




A Novel Enhanced Naïve Bayes Posterior Probability (ENBPP) Using Machine Learning: Cyber Threat Analysis

Ayan Sentuna^{1,2} · Abeer Alsadoon^{1,2}  · P. W. C. Prasad^{1,2} · Maha Saadeh³ · Omar Hisham Alsadoon⁴

Accepted: 21 October 2020 / Published online: 9 November 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract

Machine learning techniques, that are based on semantic analysis of behavioural attack patterns, have not been successfully implemented in cyber threat intelligence. This is because of the error prone and time-consuming manual process of deep learning solutions, which is commonly used for searching correlated cyber-attack tactics, techniques and procedures in cyber-attacks prediction techniques. The aim of this paper is to improve the prediction accuracy and the processing time of cyber-attacks prediction mechanisms by proposing enhanced Naïve Bayes posterior probability (ENBPP) algorithm. The proposed algorithm combines two functions; a modified version of Naïve Bayes posterior probability function and a modified risk assessment function. Combining these two functions will enhance the threat prediction accuracy and decrease the processing time. Five different datasets were used to obtain the results. Five different datasets containing 328,814 threat samples were used to obtain the processing time and the prediction accuracy results for the proposed solution. Results show that the proposed solution gives better prediction accuracy and processing time when different examination types and different scenarios are taken into consideration. The proposed solution provides a significant prediction accuracy improvement in threat analysis from 92–96% and decreases the average processing time from 0.043 to 0.028 s compared with the other method. The proposed solution successfully enhances the overall prediction accuracy and improves the processing time by solving the TTPs dependency and the prediction sets threshold problems. Thus, the proposed algorithm reaches a more reliable threat prediction solution.

Keywords Cyber threat intelligence · Deep belief network · Machine learning · Latent semantic indexing · Tactics · Techniques and procedures · Intrusion detection systems · Naïve Bayes

✉ Abeer Alsadoon
aalsadoon@studygroup.com

¹ School of Computing and Mathematics, Charles Sturt University, Sydney Campus, Sydney, Australia

² Department of Information Technology, Study Group Australia, Sydney Campus, Sydney, Australia

³ Department of Computer Engineering and Informatics, Middlesex University Dubai, Dubai, UAE

⁴ Department of Islamic Sciences, Al Iraqia University, Baghdad, Iraq

1 Introduction

1.1 Background

Detailed investigations of cyber-attacks and threats reveal the fact that attackers or organizations that perform cyber-attacks use common attack patterns to trick targets [1]. For this reason, security communities that develop defense strategies against cyber-attacks require intensive sharing and review of Cyber Threat Incident Reports (CTIR) in order to make these strategies more effective [2]. Due to the large sizes of CTIR and the addition of new attacks to concept of Advanced Persistent Threats (APT), it almost impossible for traditional methods in cyber-attack environment [1] to identify characteristic signature of the performed attacks [3]. Traditional methods for threat classification can be divided into two categories; machine learning based methods and lexicon-based methods [4]. Prior methods make calculations using lexicons to generate classification results [5]. However, reliability of results obtained with these methods depends largely on the quality and coverage of threat reports [4]. On the other hand, methods in the first category use handcrafted feature engineering to capture statistical features. With statistical data, various classifiers, such as Support Vector Machine (SVM), are used to obtain an estimated output of threat characteristics [5]. However, due to the difficulty of applying these methods, a poor performance results in classification results. To improve the performance of these methods, a combination of deep learning and machine learning methods can be used to automatically extract the features from CTIRs [6].

1.2 Review

Effectiveness and usefulness of machine learning methods in cyber threat intelligence has been proven many times. Bayesian probabilistic machine learning is based on joint probability function that graphically represents probability-based relationships between attack tactics, techniques, and procedures (TTPs). In addition, it eliminates uncertainty and provides prediction reliability in threat intelligence reports [7]. Bayesian method allows for a better extraction of threat features with low computation cost [8]. Despite all these superior features, performance of Bayesian method needs to be improved mainly in terms of the activation function, the loss function, and the parameters. For improving the processing time and prediction accuracy, optimized algorithms in threat characteristic classification can be used [7]. Models that use machine learning have improved the processing time and the classification accuracy with the help of various algorithms and techniques in determining threat classification [5]. For example, the prediction accuracy in [2] reaches 92%, while the processing time gives an average value of 0.043 s which is better compared to other models that use conventional clustering methods [7]. In addition to that, the use of Naive Bayes algorithm reduces keyword search problems and provides better performance than other models [2]. However, this algorithm assumes that all predictive threats are independent of each other [7]. This slows down the prediction stage and effecting the processing time performance, making it difficult to understand associated attacks.

1.3 Aim

The purpose of this paper is to improve the prediction accuracy and the processing time of security mechanisms against cyber-attacks. This is done by combining two functions; a modified version of Naïve Bayes posterior probability function and a modified risk assess-

ment function. The proposed modified Bayesian probabilistic graphical function [7] is used to overcome independent TTPs detection problems of current support function algorithm. Moreover, it represents the probability-based relationships between TTPs, eliminates uncertainty, and provides prediction reliability [7]. For the modified risk assessment function, this paper applies the risk assessment framework, provided in [3], for the TTP classifications. This framework [3] provides a dynamic risk management by focusing on behavioural detection of complex TTPs. By combining these modified functions, the proposed solution has solved the problems of the existing solutions and increased the classification and the prediction accuracy.

1.4 Paper Structure

The rest of this paper is organized as follows: Sect. 2 reviews the literature on current solutions of cyber threat classification using machine-learning methods. The proposed solution is discussed in Sect. 3. Section 4 discusses the experiments and results of the proposed solution. Finally, Sect. 5 concludes the paper and presents the future works. Table 1 shows the abbreviation list used in this paper.

2 Literature Review

In this section, some related papers from the literature are summarized to give better understanding of the study problem, methods, and techniques.

2.1 New Emerging Techniques, Tactics and Procedures in Cyber Threat Intelligence Reports

The solution that is proposed in [9] highlights the benefits of using high-level indicators of compromise (IoCs) in attributing cyber threats and provides a machine-learning model that supplies effective accuracy in extracting high-level indicators of compromises from unstructured cyber threat intelligence (CTI) reports. The solution enhances automated cyber threat attribution framework (FinTech) to minimize unstructured report errors in machine learning. The authors offered a solution to the problem by using distributional semantics technique and improved indexing of CTI reports. They have conducted research by integrating natural language processing into machine learning models. In addition to that, they have profiled cyber attackers and attack patterns with FinTech algorithm. Their solution provides 97% accuracy in extracting high-level IoCs from unstructured cyber-threat intelligence documents. In FinTech algorithm, semantic matching query terms are matched to terms in text corpora [1] but, synonyms and polysemous words cause false matching due to inaccurate concept matching [7]. This negatively affects latent semantic analysis during obtaining and connecting high-level attack processes [7]. As a result, the accuracy in terms of cyber threat actors is acceptable, but false matching error needs to be considered to solve the issue of matching with latent semantic query. Kim and Kim [10] enhanced cyber threat intelligence dataset that is generated from cyber intelligence reports using an automated dataset generation system CTIMiner. They aimed to increase the quality of data which can be used in cyber threat intelligence analysis techniques. They have offered a solution by using data extraction method which was enhanced by malware analysis platform that provides additional valuable collection of detailed threat data and revealing characteristics of attackers who are performing

Table 1 Abbreviation for annotations used in the report

ENBPP	Enhanced Naïve Bayes posterior probability
TTPs	Tactics, techniques and procedures
CTIR	Cyber threat incident reports
APT	Advanced persistent threats
CTI	Cyber threat intelligence
IoCs	Indicators of compromise
DRA/DRM	Dynamic risk assessment and management
OWL	Web ontology language
SWRL	Semantic web rule language
MDP	Markov decision process
LSI	Latent semantic indexing
ATT&CK	Adversarial tactics, techniques and common knowledge
BPGM	Bayesian probabilistic graphical model
IDS	Intrusion detection system
EP-ANN	Event-profile artificial neural networks
STIX	Structured threat information expression
DMS	Detection mechanism selection
TTD	TTP detection
TFS	Threat support function
LSTM	Long short-term memory
KNN	K nearest neighbor
ANN	Artificial neural networks
CRF	Conditional random fields
BPG	Bayesian probabilistic graphical model
SVM	Support vector machines
SIRS	Semantic indexer and retrieval system
TTP	Tactics, techniques and procedures
STIX	Structured threat information expression
MILP	Mixed-integer linear programming
DLNN	Deep learning neural network
CNN	Convolutional neural network
SVD	Singular value decomposition

cyber-attacks. The conducted research is done by running CTIMiner system on 612 public reports and categorizing the different types of collected data. It provides an increase of valuable data by 43%. In addition, this solution supplies high quality and structured dataset that is obtained from open sources, which provides cyber analysis techniques and statistical features suitable for CTI analysis [11]. However, quality of the results, which is obtained during parsing indicators of compromise extraction stage [12], is critically affected by parser's performance. Due to the parsers' functional limitations, there could be remaining IoCs that was not extracted from the reports database. This effects the threat analysis results and the prediction accuracy [12]. As a result, accuracy in terms of valuable threat data is accept-

able, but parser performance errors need to be considered to solve extraction issue of attack patterns [5].

Subroto and Apriyana [13] in their work have proposed statistical machine learning algorithmic model to minimize unstructured and constantly changed data errors in cyber-threat intelligence reports. In their proposed solution, they used CVE-details, R software, and dendrite/pyramid functions of Plotrix package to improve the learning confusion matrix. Their solution conducted research by integrating term document matrix [14] into CVE databases to automate statistical machine learning algorithm. Their work provides 96.73% prediction accuracy with artificial neural networks (ANN) algorithm, which analyse vulnerability patterns. With this solution a good range of prediction accuracy with better positives prediction is reached, which provides high true positive rate [15] in analysis of vulnerability patterns. In addition to that, while analysing cyber threat big data, machine-learning algorithms are used to organise and clean the data. Therefore, analysis of vulnerability patterns became more accurate. However, false-negative error is not considered [15] in this solution and lack of risk management algorithm leads to short-term information delays [14] during cyber risk analysis in CVE database. This creates an environment for false-negative errors [15]. As a result, prediction accuracy in terms of threat patterns is acceptable, but risk management algorithm needs to be considered where false-negative errors are defined. Improvement of risk calculation accuracy in security management process to minimize security events, that become an incident during cyber-threat intelligence risk assessment, is the purpose of Riesco et al. [3] study. Their solution minimises emerging threats error (also called as unknown threats error) that occurs during CTI risk management process. Authors have enhanced risk management frameworks using dynamic risk assessment and management (DRA/DRM) algorithm to minimise merged threat errors. Web ontology language (OWL) and Semantic Web Rule Language (SWRL) are used to improve operational level triggers. Authors have conducted research by integrating value added semantic algorithm format into DRM for further DRA/DRM implementation. This provides 65% risk assessment accuracy in security events. The developed dynamic risk-management framework is compatible with widely used management standards and risk assessments [16]. It also provides a degree of details and effectiveness that are required for risk management frameworks and an acceptable range of risk assessment accuracy with tactical and strategic levels of risk relationships. This supplies near real time dynamic risk assessment [16]. However, not all cyber threats that are encountered in the virtual environment were included in the risk calculation algorithm. This causes incorrect detection in real time responses [8]. As a result, prediction accuracy in terms of near real time severity gives good results, but the missing threat error need to be considered to solve the issue of detection in real time responses.

2.2 Modelling Attacker Activities Based on Close Attacks

Durkota et al. [17] developed intelligent and rational security with mathematical algorithm framework (game theory) to downgrade decision-making errors in cyber threat prediction. Their work improves accuracy of computing optimal defense strategies against complex cyber-attacks in real computer networks (multiple decision-making) environments. The offered solution uses Stackelberg equilibrium (SE), MILP formulation and Markov decision process (MDP) to improve the process of computing optimal attack policies. Authors conducted research by integrating the attack graphs into MDP algorithm to compute optimal attacker policy. This gives good results in terms of decision-making accuracy. In addition to that, the provided framework successfully found 88% optimal strategy. The proposed

solution provides an improved accuracy on decision-making process even if the attack complexity level is high. With the provided algorithm, a method which can be calculated quickly has been developed and the attack prediction rates can give high results [18]. However, the algorithm needs a large amount of processed data [4] (aggressive motivations, attack success percentages, etc.) which increases margin of error in sensitivity of decision-making strategy development [18]. As a result, the provided solution gives high results for action-success probability accuracy, but the sensitivity error needs to be considered to solve the issue of strategy development. Noor et al. [2] developed a novel machine learning based framework to minimize cyber-threat prediction errors in cyber threat intelligence. They have offered a solution to the problem by using Latent Semantic Indexing (LSI) [1] which has improved the correlated attack detection. Their work improves threat analysis process in attack prediction mechanisms that may help to identify TTPs based on artefacts that are observed with the help of appropriate machine learning algorithms. The proposed solution conducted research by ranking threat incident reports and adversarial tactics, techniques and common knowledge (ATT&CK) repositories based on historical data that measures maximal detection with novel machine learning based framework [1]. It provides attack pattern prediction accuracy of 92% and detection time of 0.04 s. This solution supplies high prediction accuracy and quite low detection time as compared to considerable time it typically takes to predict data breach incidents. This improvement provides threat analysis process using SIRS system with effective security analysis mechanism against attacks [5]. However, the algorithm assumes that all predictive TTPs are independent of each other [7]. This slows down the prediction stage making it difficult to understand associated attacks. The threat support function of the algorithm, which measures the maximal support of the detected TTPs towards a threat occurrence, tends to set all predictor TTPs as independent when function value approaches 1 or 0 [7]. This affects the model's ability to recognise attacks and reduces the overall threat prediction reliability. As a result, the prediction accuracy in terms of attack prediction is acceptable but TTPs independency needs to be considered to solve the issue of detecting unknown attacks.

Sun et al. [7] proposed a solution that enhanced the machine learning method based on Hawkes Process. The solution is for modelling attacker activities with latent distance model to effectively identify activity pattern and structure of cyber-attacks. Their work uses only temporal information without the need for complicated feature engineering. Moreover, it filters out dissimilar attacker patterns of clusters. Since the graphical clustering algorithm, that is used in the developed model, does not require prior knowledge of the number of clusters and the cluster size [2], this method has a great generality. This solution provides acceptable predictive log likelihood and it effectively models and clusters attacker activity using machine learning. The study conducted research by integrating Bayesian Probabilistic Graphical Model (BPGM) and quality-based clustering algorithm in machine learning. It provides lowest sparsity property gained by the network prior – 9.5 and effectively detect connection between attackers in large number of events. However, Gibbs sampling algorithm, that is used in the BPGM, is a time-consuming method of inference [19]. Failure to detect order of attack (attack pattern) on time arises as an important problem for security of this solution [2]. As a result, predictive log likelihood in terms of attacker activity is acceptable, but cluster size needs to be considered to solve algorithm performance issues.

2.3 Advanced Malware Prediction with Regression Models

Husak et al. [6] have enhanced the attack prediction using attack projection and intention recognition algorithm to minimize prediction mistakes in intrusion detection system (IDS). In their study, the authors used artificial neural network and support vector machine (SVM) machine learning algorithms to improve attack prediction. They conducted research by integrating data mining and neural networks into intrusion detection system to reduce the complexity and learning time of prediction algorithm [13]. Depending on the length of the applied attack scenario, the proposed work provides 92.3–99.2% accuracy rate. This solution provides a good range of accuracy with a minimal time delay, which supplies prediction for even very specific attacks [12]. Attack prediction accuracy has been increased since data mining has been added to the machine-learning algorithm for learning and generating attack models or attack plans [20]. However, the loss function, which causes small changes in the beginning of attack prediction in SVM algorithm, causes a slowdown in prediction of attacks that use different models [20]. As a result, the accuracy in terms of frequency of mistakes is acceptable, but automated attack plan library generation needs to be considered to solve the issue of prediction changes. Lee et al. [5] aimed to improve capacity of deep learning-based methods to transform security incidents that are collected into individual activities to prevent advanced cyber threats. For this reason, to minimize false positive errors, they developed artificial intelligence security information and event management cyber-threat detection technique (AI-SIEM). They offered a solution to the problem by using large-scaled event profiles and deep learning detection methods has enhanced accuracy performance. The integration of term frequency-inverse document frequency (TF-IDF) indexing mechanism for very large scale of data in AI-SIEM algorithm improves true positive accuracy. Overall best accuracy was delivered by the proposed event-profile artificial neural networks (EP-ANN) models with accuracy score of 0.93–0.99 in four experiment datasets in cyber-threat intelligence analysis. This work provides an acceptable improvement of true positive accuracy with rapidly respond time, which supply cyber-threat detection ability in large-scale cyber security environment. AI-SIEM system quickly and effectively compares long-term security analysis [21] and highlights important security alerts, therefore, false positive alerts are reduced [6]. AI-SIEM algorithm yield very good results on benchmark datasets, but accuracy inconsistencies are observed with application of the system in EP-ANN algorithm [21]. As a result, accuracy in terms of true positive and false positive is accepted, but rare attack data learning algorithm needs to be considered to solve application of system in EP-ANN algorithm issue.

Bahtiyar et al. [20] enhanced advanced malware prediction with multi-dimensional machine learning technique to downgrade malware detection errors. The proposed solution uses linear, polynomial, and random forest regression models [22] to improve correlation value. This solution conducted research by the integration of regression algorithms into correlations among features and provides 0.8203 extracted correlation value between advanced malware software features and 0.558 closeness rate to advanced malware. The study provides improved prediction accuracy and efficiency with extracted closeness score and correlation value. This supplies certain identification in advanced malware prediction. Machine learning approach uses correlations among five features and four regression algorithms to predict advanced malwares [4]. In this study, random forest regression four feature has yielded better results on analysis, but an acceptable threshold value has not been achieved in precise definition of advanced malware software [8]. This mean that advanced malware features dependencies are not taken into account [4]. Therefore, machine-learning datasets, which contain newly founded advanced malicious software samples, should be added to multi-dimensional algorithm. Moreover, a threshold value should be entered into the system as

fixed value not as random value [8]. As a result, the accuracy in terms of malware features is acceptable, but fixed value needs to be considered to solve this issue in precise definition of advanced malware software.

2.4 State of Art

In this part, system's features, which are highlighted inside blue broken line in Fig. 1, and limitations, which are highlighted inside red broken line in Fig. 1, are presented. Noor et al. [2] proposed an enhanced novel machine learning based framework algorithm to minimize cyber-threat prediction errors. The use of Latent Semantic Indexing (LSI) has improved the correlated attack detection. This study conducted research by ranking threat incident reports (CTIR) and adversarial tactics, techniques and common knowledge (ATT&CK) repositories based on the historical data in order to measure maximal detection with novel machine learning based framework [2]. It provides attack pattern prediction accuracy of 92–100 and detection time of 0.04 s. This model consists of three stages shown in Fig. 1 which are 1- semantic indexer and retrieval system stage, 2-TTD semantic network stage, and 3- cyber threat prediction stage.

Stage 1 Semantic Indexer and Retrieval System (SIRS)

Cyber threat incident reports and adversarial tactics, techniques and common knowledge documentations are the inputs of the system. While a CTIR corresponds to a single cyber threat, an ATT&CK document may correspond to several detection mechanisms associated with a TTP. To build threat TTP Detection (TTD) network, TTPs are extracted from structured threat information expression (STIX) and encoded as cyber-threat incident reports. After that, TTPs dictionary is built [2]. In this stage, a second step is to make every single TTP semantically correlated with malware attacks in the CTIR and with TTPs in ATT&CK documents. In order to connect TTPs with detection mechanism, instead of using simple keyword search, ranking process is applied with the help of LSI to CTIR and ATT&CK for each TTPs present in TTD [2].

Stage 2 TTD Semantic Network Stage

In this stage, threats are linked to their TTPs and detection mechanisms. In order to represent semantic relations of TTPs under three specific concept, the detection mechanism set, threat set, TTP set for cyber threat reports, ranked cyber threat incident reports, and ATT&CK threats are linked to their relative TTPs and detection mechanisms [2]. Then, to predict threats based on the existence of determined artefacts, a network of probable events is trained between threats and TTPs based on historical data to measure the maximal support of detected TTPs towards a threat occurrence. *Limitation* Experiment results illustrates that the state of art solution even in the worst case scenario where TTPs have high level overlap, has achieved prediction of attack pattern accuracy average of 92% and in ideal situation threat prediction accuracy becomes %100. However, with this model, threat support function in threat TTP detection semantic network algorithm assumes that all predictive TTPs are independent of each other [7]. This slows down prediction stage [7] making it difficult to understand the associated cyber-attacks [5]. *Limitation Justification* the threat support function of the algorithm, which measures the maximal support of detected TTPs towards a threat occurrence, tends to set all predictor TTPs as independent when function value approaches 1 or 0. This affects the model's ability to recognize attacks and reduces the overall threat prediction reliability [5].

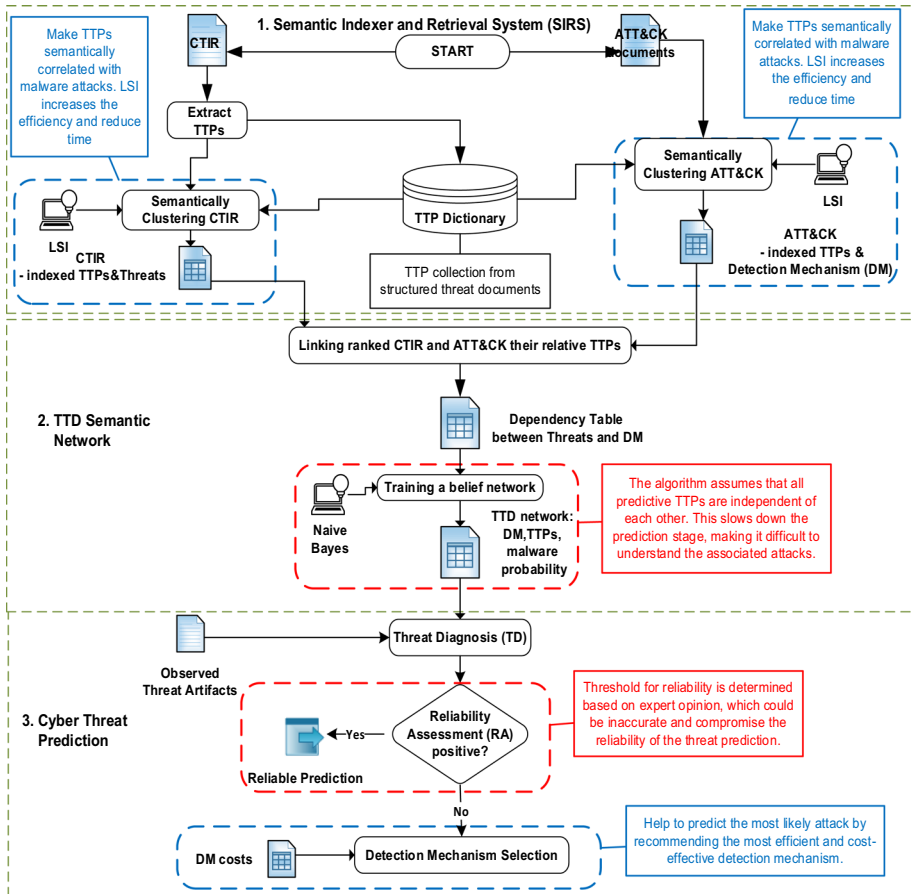


Fig. 1 Block diagram of state of art system [2]. Good features of state of art are shown inside blue broken lines and limitations of state of art are shown inside red broken lines

Stage 3 Cyber Threat Prediction Stage

The first step of this stage is the threat investigation (TD) module. The responsibility of this step is to produce a predicted threat based on the detected TTPs in order to predict a set of threats. Next step is reliability assessment (RA) in which reliability of prediction is measured. In case of high reliability, threat investigation is considered completed. Otherwise, a set of existing TTPs are considered by detection mechanism selection. Therefore, RA step reduces time and resource consumption by minimizing likelihood of incorrect prediction caused by the prediction with low reliability and presence determination of TTPs [2]. Last step of this stage is detection mechanism selection (DMS). This step is needed to help the cyber-security analyst to investigate threat artefacts against most likely attack family. This is done by recommending the most efficient and cost-effective detection mechanism a set of existing TTPs linked with detection mechanism is calculated based on cost efficiency. In case of sufficient reliability grow, the prediction is terminated. Otherwise, the TD receives set of predicted TTPs [2]. *Limitation* There is a problem in the prediction sets which give highest probability of occurrence values for classification of detected TTPs. This problem arises as

to which malware instances should be included in prediction sets [3]. *Limitation Justification* When constructing prediction sets correctly, it is necessary to determine a threshold to reach more reliable threat prediction values. Expert opinion is used to determine this threshold which could be inaccurate and compromise the reliability of threat prediction [2].

This model presented prediction of attack pattern accuracy of 92–100%, prediction reliability in terms of number of detected TTPs of 50–60% and prediction threat incidents for detected TTPs in an average time of 0.04 s [2].

To link each threat in the dependency table to its respective TTPs and detection mechanism, a normalized table is used and normalized probability or normalized conditional probability is calculated. Normalized posterior probability defines the objective function, which is computed by Naïve Bayes technique as shown in Eq. (1) [2]. However, this slows down the prediction stage making it difficult to understand the associated attacks and downgrade the prediction reliability.

$$\mu(t_i|ttp_i) = \frac{\omega(ttp_i|t_i)p(t_i)}{\sum_{t_i \in T_{tpp_i}} \omega(ttp_i|t_i)p(t_i)} \tag{1}$$

where $\mu(t_i|ttp_i)$: normalized posterior probability using Naïve Bayes; $\omega(ttp_i|t_i)$ normalized conditional probability; $p(t_i)$: prior class probability; t_i : threat in dependency table built between threat incidents and TTPs; T_{tpp_i} : set of detected TTPs; $p(ttp_i|t_i)$: conditional probability between TTPs and threats; $ttp_i|t_i$: event ttp_i given event t_i .

The threat set that is associated with the detected TTPs from dependency table is built, and historical threat occurrences, threat likelihood, and threat set prior probability are obtained to build Bayesian probabilistic graphical model for the threat probability estimation. Normalized conditional probability is calculated using Eq. (2) [2].

$$\omega(ttp_i|t_i) = \frac{p(ttp_i|t_i)}{\sum_{t_i \in T_{tpp_i}} p(ttp_i|t_i)} \tag{2}$$

where $\omega(ttp_i|t_i)$: normalized conditional probability; $p(ttp_i|t_i)$: conditional probability between TTPs and threats; ttp_i : detected TTPs; t_i : threat in dependency table; $ttp_i|t_i$: event ttp_i given event t_i ; T_{tpp_i} : detected TTPs set.

Belief network is implemented in TTD semantic network stage. A network of probable events is trained between threats and TTPs based on historical data to measure the maximal support of detected TTPs towards a threat occurrence. To show maximal support, threat support function is calculated as shown in Eq. (3) [2]. However, accuracy and prediction reliability can be increased by techniques for best alignment.

$$S(t_i) = \frac{\sum_{ttp_i \in TTPD_i} \mu(t_i|ttp_i)}{\sum_{ttp_i \in TTP_i} \mu(t_i|ttp_i)} \tag{3}$$

where $S(t_i)$: threat support function; $TTPD_i$: set of detected TTPs due to threat t_i ; TTP_i : TTPs set, associated with a threat t_i ; $\mu(t_i|ttp_i)$: normalized posterior probability using Naïve Bayes; $ttp_i|t_i$: event ttp_i given event t_i ; t_i : threat in dependency table; ttp_i : detected TTPs (Table 2).

3 Proposed System

In cyber threat intelligence environment, machine learning algorithms which use different feature extraction and classification techniques has been analysed in detail, pros and cons of each method have been determined. After analysis, it has been found that accuracy, reliability,

Table 2 Belief network algorithm

<p>Algorithm: Belief network to measure maximum support of detected TTPs Input: Linked threats and TTPs (T_{tpi}) Output: TTD network with detected threats</p>
<p>BEGIN</p> <p>Step 1: Get threat t_i</p> <p>Step 2: Check the dependency table ($p(tp_i t_i)$). If the threat t_i is associated with the detected set of TTPs (T_{tpi}) then go to Step 3, otherwise go back to Step 1.</p> <p>Step 3: Calculate conditional probability $p(tp_i t_i) \in (0, 1)$ based on historical threat occurrences. Go to Step 4.</p> <p>Step 4: If null values from step 3, then normalize value by adding 1 to all entries of the frequency table and go to Step 6. Otherwise go to Step 5.</p> <p>Step 5: Calculate normalized conditional probability ($\omega(tp_i t_i)$) and go to Step 6.</p> <p>Step 6: Calculate normalized posterior probability $\mu(t_i tp_i)$ using prior class probability ($p(t_i)$) and go to Step 7.</p> <p>Step 7: Consider all T_{tpi} to calculate the maximal support of TTPD; towards t_i and go to Step 8.</p> <p>Step 8: If threat t_i for the given TTP; has the maximum posterior probability, then add threat t_i to the prediction set (TTD network) and exit. Otherwise, go back to Step 1.</p> <p>END</p>

detection time, and false detection are key factors that impact threat prediction neural network algorithm. Noor et al. [2] was selected as the state of the art for the proposed solution in this paper among other collected and analysed methods. The main reason behind this selection was the proposed novel machine learning based framework for cyber-attack prediction. Novel machine learning based technique semantically extracts threats and attack tactics, techniques, procedures from known threat sources to create a semantic network [2]. Semantic network establishes probability relationships between threats and TTPs using Naïve Bayes machine learning to identify and predict threats. Naïve Bayes computing normalises conditional probability of threat TTP mapping and therefore, finds a best candidate threat prediction set. In addition to that, to enhance the prediction accuracy, a novel machine learning based technique is combined with the belief network model [2]. However, this work has several limitations. One limitation is that the threat support function of the algorithm, which measures the maximal support of detected TTPs towards a threat occurrence, tends to set all predictor TTPs as independent when function value approaches 0 or 1. This affects the model's ability to recognise attacks and reduces the overall threat prediction reliability. Moreover, it slows down prediction stage, making it difficult to understand associated cyber-attacks. Another limitation is that there is a problem in prediction sets which give the highest probability of occurrence values for classification of detected TTPs. This problem arises as to which malware instances should be included in prediction sets. To overcome independent TTP detection problem of support function algorithm, Bayesian probabilistic graphical model that is based on joint probability function inspired by Sun et al. [7] was used. This model graphically represents probability-based relationships between TTPs, eliminates uncertainty, and provides prediction reliability. Another new feature of the proposed solution is the application of risk assessment framework that was proposed by Riesco et al. [3] in TTP classifications. This framework provides dynamic risk management by focusing on behavioural detection of complex TTPs. Application of these new features has solved problems of existing solution, increased classification and prediction accuracy, and reduced processing time.

The proposed system consists of three major stages (Fig. 2) which are: (1) semantic indexer and retrieval system (SIRS), (2) TDD semantic network, and (3) cyber threat prediction.

Stage 1 Semantic Indexer and Retrieval System (SIRS)

This stage follows architecture of Noor et al. [2] solution where threat incident reports and adversarial tactics, techniques and common knowledge documentations are the input of the system, see Fig. 2. While a CTIR corresponds to a single cyber threat, an ATT&CK document

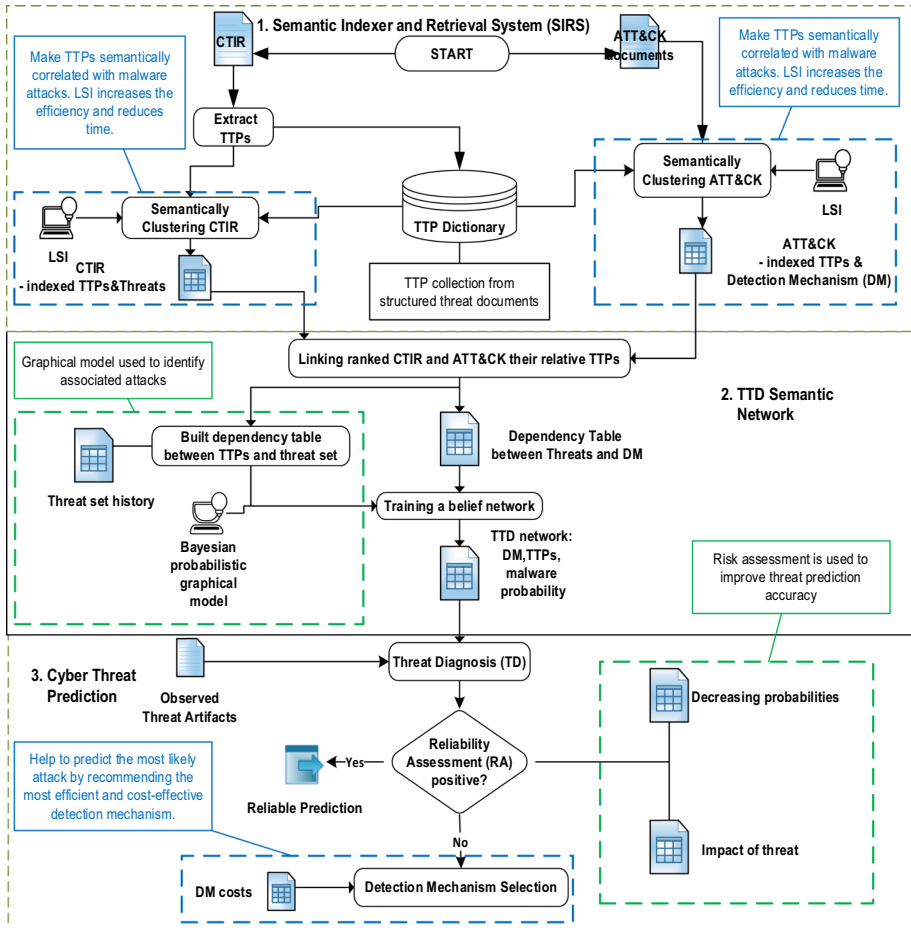


Fig. 2 Block diagram of the proposed threat prediction method using Bayesian probabilistic graphical algorithm. Green borders refer to the new parts in the proposed system

may correspond to several detection mechanisms associated with a TTP. As shown in Fig. 2, to build threat TTP detection network, TTPs are extracted from structured threat information expression and encoded as cyber threat incident reports. After that, TTPs dictionary is built. Instead of using simple keyword search, ranking process is applied with the help of LSI to CTIR and ATT&CK for each TTPs present in TTD. Therefore, every single TTPs semantically correlated with malware attacks in CTIR and with TTPs in ATT&CK documents TTPs with detection mechanism connected (see Fig. 2).

Stage 2 TTD Semantic Network

In the second stage of the proposed model as shown in Fig. 2, in order to represent semantic relations of TTPs under three specific concepts (detection mechanism set, threat set, and TTP set for cyber threat reports), ranked cyber threat incident reports and adversarial tactics, techniques and common knowledge threats are linked to their relative TTPs. After that, to predict threats based on existence of determined artefacts, Bayesian probabilistic graphical model is used to identify associated attacks [7]. Historical threat occurrences, threat likelihoods, and

threat set probabilities are obtained to build the Bayesian probabilistic graphical model [7], see Fig. 2. In addition to this step, joint distribution threat posterior probability is calculated to build a dependency table between TTPs and threat set. Therefore, the algorithm’s threat probability estimation and normalised conditional probability are improved. Compared to threat support function, graphical model is simpler and solves dependency problem of TTPs. As a next condition, a network of probable events is trained between threats and TTPs based on historical data to measure maximal support of detected TTPs towards a threat occurrence.

Stage 3 Cyber Threat Prediction

To predict a set of threats, the responsibility of this step is to produce a predicted threat based on the detected TTPs. Next, the reliability of prediction is measured. In case of high reliability, threat investigation is completed. Otherwise, a set of existing TTPs are considered by detection mechanism selection, see Fig. 2. Therefore, this step reduces time and resource consumption by minimizing the likelihood of incorrect prediction caused by the prediction with low reliability and determination of the presence of TTPs. At this stage, the proposed work applies a dynamic risk management framework (see Fig. 2). This framework assesses the risk using threat impact and decreasing values of probability due to the implemented and new proposed measures [3]. Therefore, all threat sets of detected TTPs can be considered to assess the maximal support of a set of detected TTPs towards dependency table [3]. As a result, threat set with maximum posterior probability can be considered as predicted threat set. This framework solves prediction sets threshold problem and helps the algorithm to reach more reliable threat prediction level. To help investigation of threat artefacts against most likely attack family by recommending efficient detection mechanism, a set of existing TTPs linked with detection mechanism is calculated (see Fig. 2). In case of sufficient reliability grow, the prediction is terminated. Otherwise, threat diagnosis receives a set of predicted TTPs.

3.1 Proposed Equation

Identify prediction sets that induce observed data and capture distributions that characterize relationships between hidden states and hidden variables are critical to threat prediction. Bayesian probabilistic graphical model, that is based on joint distribution, is used to calculate posterior probability to avoid the problem arises as to which malware instances should be included in prediction sets. It increases probability accuracy due to associated threats consideration compared to Posterior Naïve Bayes probability that is based on normalised conditional probability. Joint distribution is defined as in Eq. (4) [7].

$$p(t_i, T_{tppi} | ttp_i) = p(ttp_i | t_i, T_{tppi}) p(t_i | T_{tppi}) p(T_{tppi}) \tag{4}$$

where $p(t_i, T_{tppi} | ttp_i)$: joint distribution; t_i : threat in threat incidents and TTPs dependency table; ttp_i : detected TTPs; T_{tppi} : detected TTP threat set; $p(t_i | T_{tppi})$: likelihood t_i if T_{tppi} ; $p(T_{tppi})$: prior T_{tppi} probability.

Historical artifacts, which show presence of cyber-attack, are used to calculate the probability between threats and TTPs. To configure probability, historical data that make up frequency tables are used for TTP—threat mapping. Accordingly, it is necessary to normalise the frequency table in order to avoid null values. History probability of threat for the detected threat set ($p(t_i | T_{tppi})$) is used to find threat that is associated with a certain threat set and detected TTPs [7]. Therefore Eq. (4) is modified by us in Eq. (5).

$$Mp(t_i) = p(t_i | T_{tppi}) p(T_{tppi}) \tag{5}$$

where $Mp(t_i)$: modified prior class probability; t_i : threat in dependency table; ttp_i : detected TTPs; T_{tppi} : detected TTP threat set; $P(t_i|T_{tppi})$: history likelihood for threat set; $P(T_{tppi})$: prior T_{tppi} probability.

For a predictive nature, malware class TTP with the highest posterior probability is considered as predicted output. Similarly, among all TTPs, TTD network estimates threat event, host and network artifacts using symptoms to calculate the highest probability. Based on this, with prior threat class probability, the enhanced equation is used to get threat posterior probability. Therefore, Eq. 1 is modified by us to be in Eq. 6.

$$M\mu(t_i, T_{tppi} | ttp_i) = \left(\frac{\omega(ttp_i | t_i)}{\sum_{t_i \in T_{tppi}} \omega(ttp_i | t_i)} \right) Mp(t_i) \tag{6}$$

where $M\mu(t_i, T_{tppi} | ttp_i)$: modified version of Naïve Bayes posterior probability; t_i : threat in dependency table built between threat incidents and TTPs; ttp_i : detected TTP. T_{tppi} : Threat set of detected TTP; $\omega(ttp_i | t_i)$: normalized conditional probability.

The risk assessment approach, which is used to identify most relevant threats in the threat set, increases the accuracy of probability function and reduces the time for threat prediction. Posterior probability threat risk assessment is performed using threat impact, decreasing values of probability, and the new proposed measures as given in Eq. 7 [3].

$$R_{ti} = P_{ti} + I_{ti} - C1_{ti} - C2_{ti} \tag{7}$$

where R_{ti} : risk of threat t_i ; P_{ti} : probability of threat t_i ; I_{ti} : impact of threat t_i ; $C1_{ti}$: decreasing value of probability P_{ti} ; $C2_{ti}$: decreasing value of probability P_{ti} .

To consider treat probability after risk assessment and to assess maximum support of a set of detected TTPs (depending on security events and time), threat support function is used. Since risk assessments may be updated dynamically, risk management treatments and classification may also be updated automatically. Therefore Eq. 7 is modified by us in Eq. 8.

$$MR_{ti} = I_{ti} - C1_{ti} - C2_{ti} \tag{8}$$

where MR_{ti} : modified residual risk assessment; I_{ti} : impact of threat t_i then it materialized; $C1_{ti}$: decreasing value of probability P_{ti} due to implemented measures; $C2_{ti}$: decreasing value of probability P_{ti} due to new modified measures.

The threat support function defines the best candidate threat prediction set (prediction sets which give highest probability of occurrence values for classification of detected TTPs) with maximum probability value [21]. Therefore, Eq. 3 is enhanced by us to propose Eq. 9.

$$ES(t_i) = \frac{\sum_{tppi \in TTPD_i} M\mu(t_i, T_{tppi} | ttp_i)}{\sum_{tppi \in TTP_{Pi}} M\mu(t_i, T_{tppi} | ttp_i)} + MR_{ti} \tag{9}$$

where $ES(t_i)$: enhanced Naïve Bayes posterior probability; detected TTPs set ($TTPD_i$) $ttp_i = \{ttp_1, ttp_2, ttp_3, \dots, ttp_n\}$; detected threat $t_i = \{t_1, t_2, t_3, \dots, t_n\}$ in detected threat set T_{tppi} ; TTP_{Pi} : TTPs set associated with threat t_i . $TTPD_i$: Detected TTPs set due to threat t_i ; $M\mu(t_i, T_{tppi} | ttp_i)$: modified version of Naïve Bayes posterior probability; MR_{ti} : modified residual risk assessment.

3.2 Area of Improvement

In this solution, two equations were proposed, and the current method performance was improved. First, threat support function is modified to calculate best candidate threat prediction set with maximum probability value as shown in Eq. (3). It uses Bayesian probabilistic

graphical model which is based on joint distribution to calculate posterior probability. The purpose of posterior probability is to create a dependency table between TTPs and threat set. Therefore, dependency table with the help of prior threat class-probability calculation is used to find threat that is associated with a certain threat set which gives the best threat class probability. This solves dependency problem of the function and improves the threat prediction accuracy. Second, dynamic risk management function is combined with cyber threat prediction. With the help of Eqs. (8 and 9), posterior probability threat risk assessment is performed using threat impact and decreasing values of probability. The purpose of risk management is to assess the maximal support of a set of detected TTPs towards dependency table and to consider as predicted threat set with maximum posterior probability. This helps the algorithm to provide more reliable threat prediction and improves processing time.

Why enhanced Naïve Bayes posterior probability: Naïve Bayes posterior probability emphasizes that security incidents can be matched to tactics mapped to artificial objects in such a way that machines can identify these possibilities with certain possibilities. Using modified threat support function (TSF) as an activation function in the threat prediction algorithm, effectively avoids dependency problem of TTPs. It uses threat set associated with detected TTPs, threat likelihood, threat set prior, and historical threat occurrence as input values. The proposed work can solve dependency problem as graphical model of the Bayesian probabilistic effectively find best threat class probability. Appropriate matching of threat classes enhances prediction performance. In addition, Naive Bayes is simpler and faster than other algorithms, resulting in a more effective training process. Moreover, the proposed study considers risk management during threat prediction phase. Risk management framework considers treat probability after risk assessment to assess the maximum support of a set of detected TTPs towards threat using enhanced TSF. This improves the overall performance and enhances the prediction accuracy and processing time.

Independent detection of TTPs by threat-TTP-detection algorithm makes it almost impossible to detect associated threats while slowing down prediction stage. This affects the algorithm's ability to recognize attacks and reduces the overall threat prediction reliability. Therefore, the proposed work provides precise analysis of related threats using Posterior Naive Bayes based on normalised conditional probability, which increases probability accuracy. In addition to that, in prediction algorithms, lack of risk management during threat prediction phase reduces the overall classification performance. This becomes a problem in prediction sets, which give the highest probability of occurrence values for classification of detected TTPs. The proposed work considers treat probability after risk assessment to generate maximum support results of detected TTPs. This effectively prevents threshold mistakes in prediction sets and enhances reliable threat prediction values.

The threat support function that is used as an activation function in the state of art system faces dependency problem of TTPs. The proposed study solved this problem with modified threat support function based on Bayesian probabilistic graphical model. Moreover, there have been prediction reliability problems in the state of art system since risk assessment during threat prediction stage is not performed. The proposed work addresses this problem by including dynamic risk management framework in the prediction algorithm (Fig. 3, Table 3).

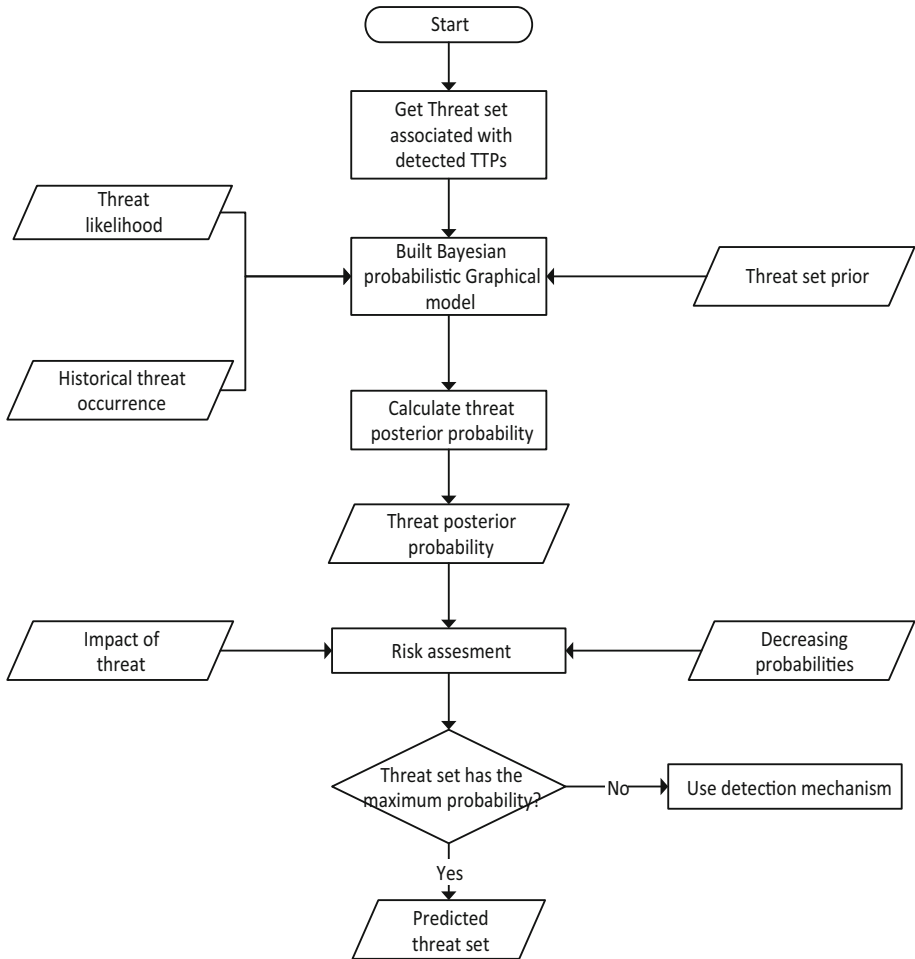


Fig. 3 Flowchart of the proposed modified Bayesian probabilistic graphical model for threat prediction

4 Results and Discussion

In this research, Python version 3.6.9, Scikit Learn, Matplotlib, Keras, Tensorflow, and Numpy libraries were used in the application and test stages. Five datasets are used from NSL-KDD, CICIDS2017 [5], and ATT&CK [2]. These datasets are publicly accessible and free. In total, the number of records of each dataset is different; the specifications are given in detail in Table 4. Data is divided into five different sets. One of them was used for test purposes and the remaining sets were used for training purposes (Figs. 4 and 5). These procedures were performed by applying holdout cross-validation. The used system configuration for experiment is as follows: Intel® Core™ I7–8550U CPU @ 1.80 GHz and 16 GB installed memory (RAM). Python 3.6.9 Keras library *Metric method* was used to calculate the prediction accuracy and the processing time values of the five different datasets. In addition, average prediction accuracy and average processing time values were calculated using *Mean method* of Python 3.6.9 Numpy library. Figures 6 and 7 show results of the different datasets.

Table 3 Proposed Bayesian probabilistic graphical model (BPGM) algorithm

<p>Algorithm: Bayesian probabilistic Graphical model (BPGM) to measure the maximum support of the detected TTPs.</p> <p>Input: Threat set associated with detected TTPs, threat likelihood, threat set prior, historical threat occurrence.</p> <p>Output: Predicted threat set with maximum risk probability.</p>
<p>BEGIN</p> <p>Step 1: Get threat set (T_{tppi}) associated with detected TTPs from dependency table ($p(tp_i/t_i)$) and go to Step 2.</p> <p>Step 2: Build Bayesian probabilistic graphical model for threat probability estimation based on the historical threat occurrences $p(tp_i/t_i)$, threat likelihood $P(t_i T_{tppi})$, and the threat set prior probability $P(T_{tppi})$. Go to Step 3.</p> <p>Step 3: Calculate the normalized conditional probability $\omega(tp_i t_i)$ using $p(tp_i/t_i)$ and go to Step 4.</p> <p>Step 4: Calculate threat posterior probability $\mu(t_i, T_{tppi} tp_i)$ using joint distribution and go to Step 5:</p> $M\mu(t_i, T_{tppi} tp_i) = \left(\frac{\omega(tp_i t_i)}{\sum_{t_i \in T_{tppi}} \omega(tp_i t_i)} \right) Mp(t_i)$ <p>Step 5: Assess the risk using threat impact I_i and the decreasing values of probability P_i due to implemented and new proposed measures $C1_i, C2_i$. Go to Step 6.</p> <p>Step 6: Consider all T_{tppi} to assess the maximal support of a set of detected TTPs towards t_i (TTPD$_i$). Go to Step 7.</p> <p>Step 7: If the threat set T_{tpp} has the maximum posterior probability, then consider it as predicted threat set and exit. Otherwise, use detection mechanism to evaluate the threat set.</p> <p>END</p>

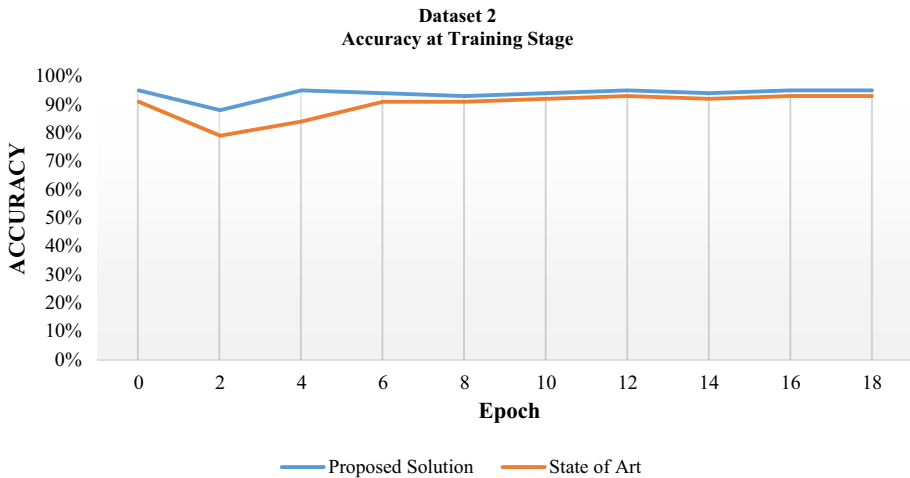


Fig. 4 Dataset 2 classification accuracy for state of art and the proposed solution. **a** Orange line indicates classification accuracy values of state of art study [2]. **b** Blue line indicates classification accuracy of the proposed solution

Classification accuracy values for the different TTP types, that are available in the used datasets, were obtained using *Predict method* of Python 3.6.9 Keras library. Processing time values have been calculated again with the help of Python 3.6.9 Keras library *Now method*. Microsoft Excel functions are used to calculate average processing time and accuracy values. All results can be seen in Figs. 8, 9, 10, 11, 12, 13, 14, 15, 16 and 17.

Belief network model was created during feature extraction and classification stages. Using this model, features that are obtained from training data are extracted. Features that make up feature maps are considered by belief network in linear time [23]. After this, posterior

Table 4 Number of records of each datasets

Dataset	Number of records										Total
	Catchamas	Duqu	Kazuar	Lazarus Group	Machete	NetTraveler	PowerDuke	Remexi	Soundbite	Winerack	
Dataset1	13,449	9234	2289	11	209	3308	46	612	6951	113	36,222
Dataset2	67,343	45,927	11,656	52	995	8745	245	923	51	877	136,814
Dataset3	9711	7458	2421	200	2654	631	92	1002	846	23	25,038
Dataset4	23,211	11,258	3687	168	1980	1134	147	2020	3278	739	47,622
Dataset5	33,201	28,546	4218	39	726	9981	302	1640	4453	12	83,118

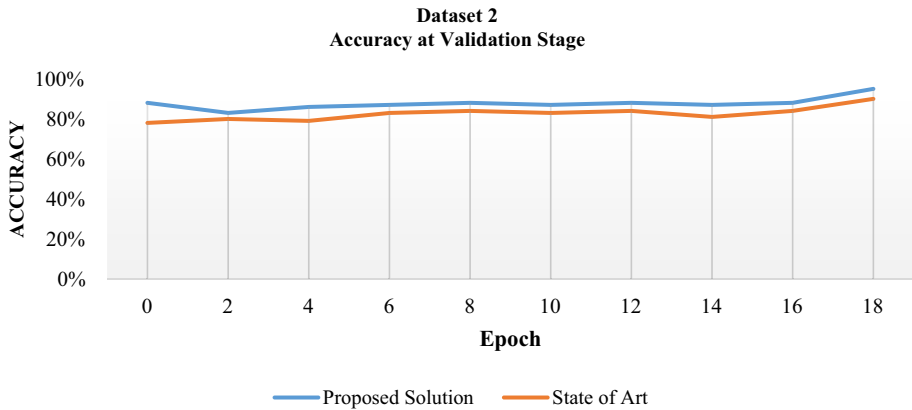


Fig. 5 Dataset 2 classification accuracy for state of art and the proposed solution. **a** Orange line indicates classification accuracy values of state of art study [2]. **b** Blue line indicates classification accuracy of the proposed solution

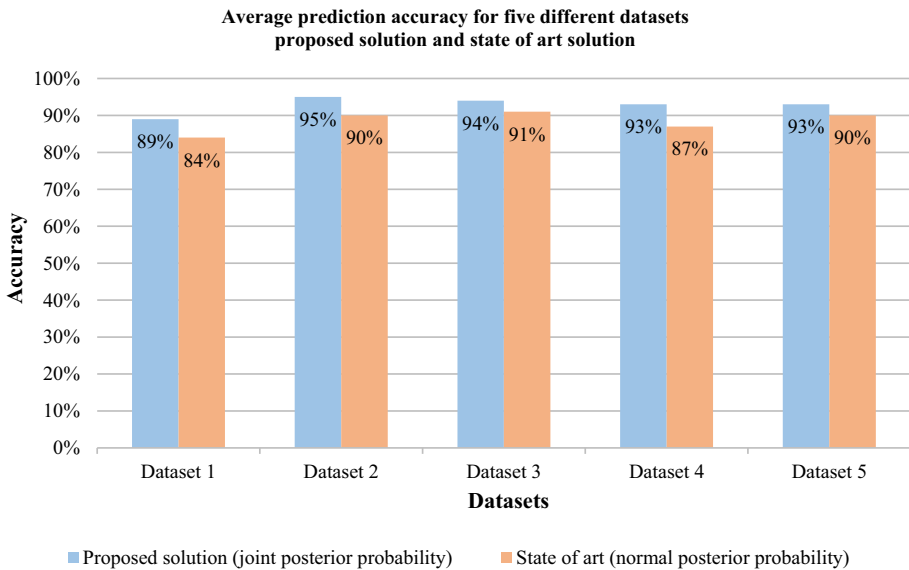


Fig. 6 Average prediction accuracy calculated for five different datasets (in percentage). Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average accuracy of Dataset (1). **b** Second two lines show average accuracy of Dataset (2). **c** Third two lines show average accuracy of Dataset (3). **d** Fourth two lines show average accuracy of Dataset (4). **e** Fifth two lines show average accuracy of Dataset 5

probabilities are calculated using support function to generate the classification of TTPs based on incident frequency. After the training process, the model was evaluated with the use of validation dataset.

The classification accuracy performance of Dataset 2 is shown in Fig. 4. Results compares the values of state of art system [2] and the proposed solution during the training phase. State of art system and the proposed solution have similar accuracy values. However, in order to

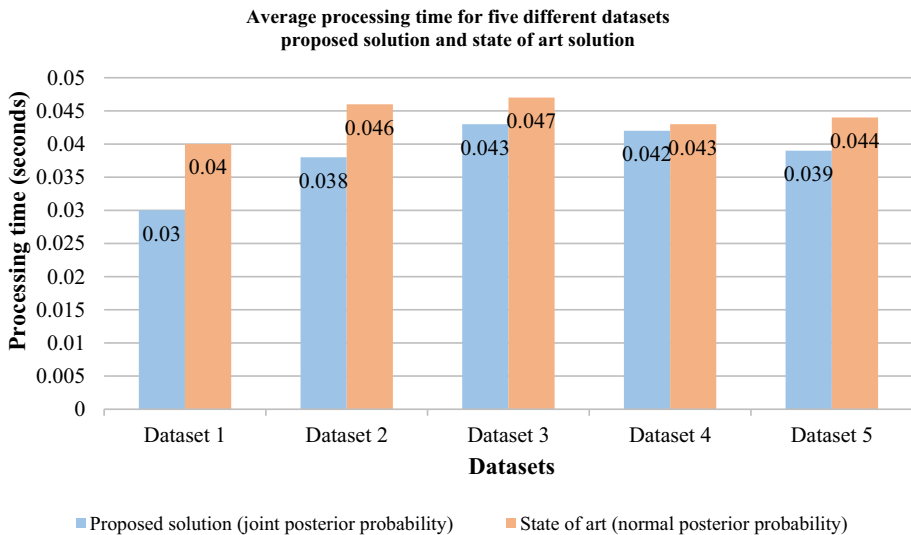


Fig. 7 Average processing time calculated for five different datasets (in seconds). Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average processing time of Dataset (1). **b** Second two lines show average processing time of Dataset (2). **c** Third two lines show average processing time of Dataset (3). **d** Fourth two lines show average processing time of Dataset (4). **e** Fifth two lines show average processing time of Dataset 5

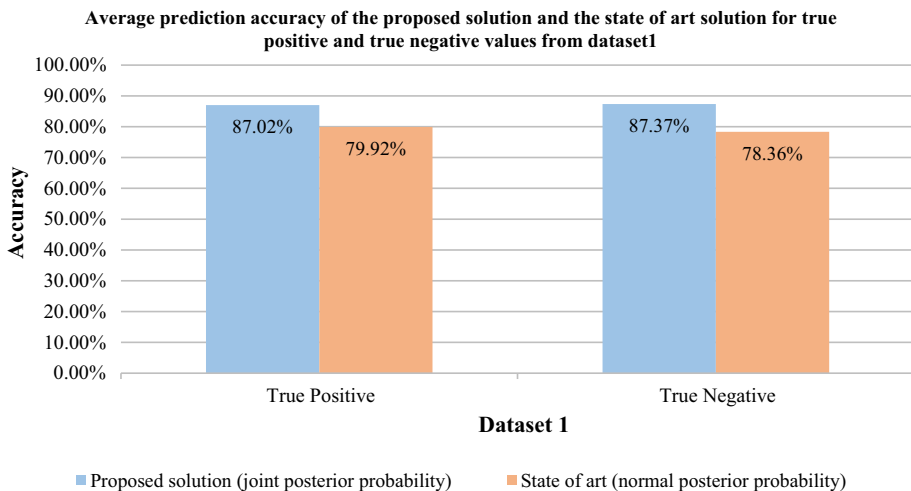


Fig. 8 Average prediction accuracy calculated for true positive and true negative results (in percentage) from Dataset 1. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average accuracy of true positive. **b** Second two lines show average accuracy of true negative

achieve optimum accuracy values, less epochs are taken by the proposed solution, it means that in case of increasing dataset size, the proposed solution will reduce the processing time values in model training process.

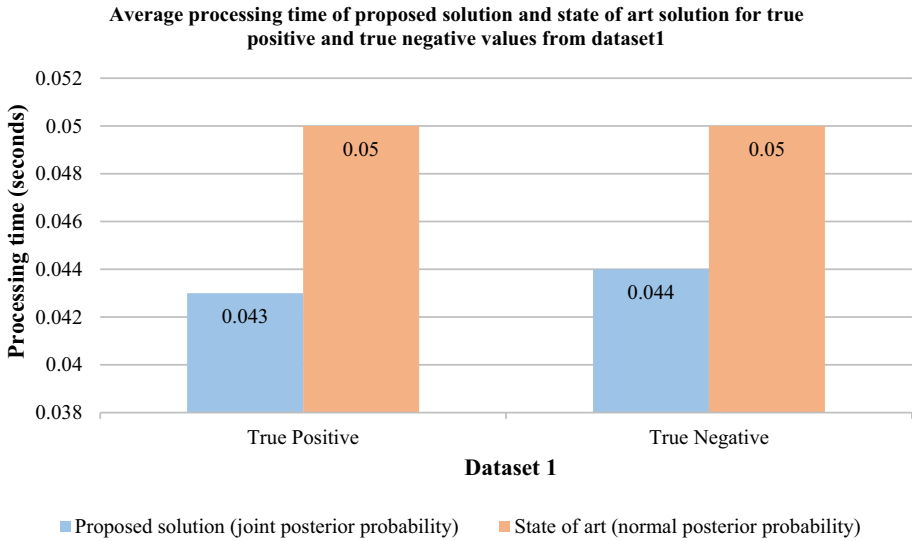


Fig. 9 Average processing time calculated for true positive and true negative results (in seconds) from Dataset 1. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average processing time of true positive. **b** Second two lines show average processing time of true negative

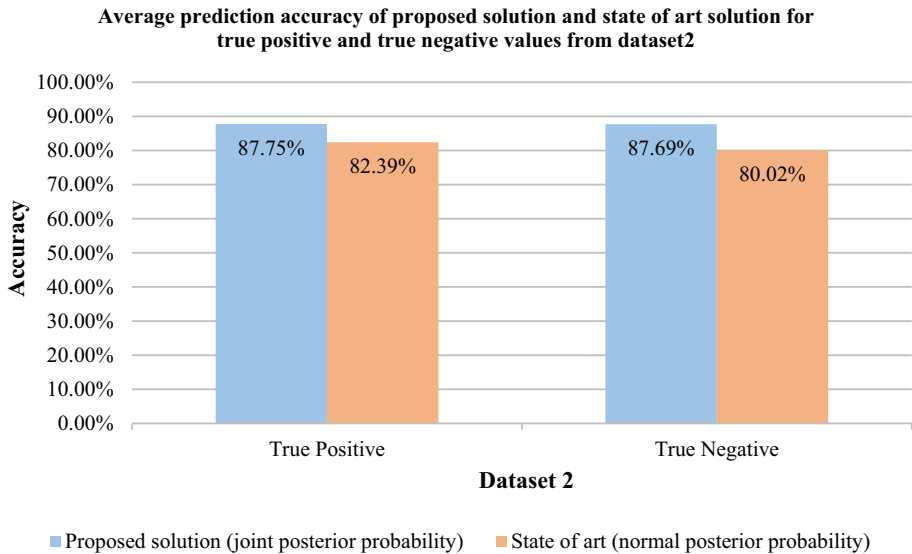


Fig. 10 Average prediction accuracy calculated for true positive and true negative results (in percentage) from Dataset 2. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average accuracy of true positive. **b** Second two lines show average accuracy of true negative

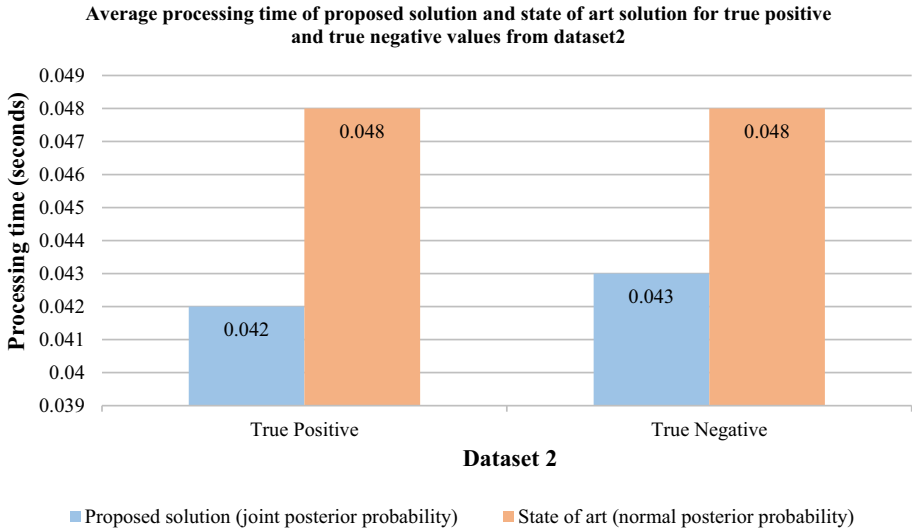


Fig. 11 Average processing time calculated for true positive and true negative results (in seconds) from Dataset 2. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average processing time of true positive. **b** Second two lines show average processing time of true negative

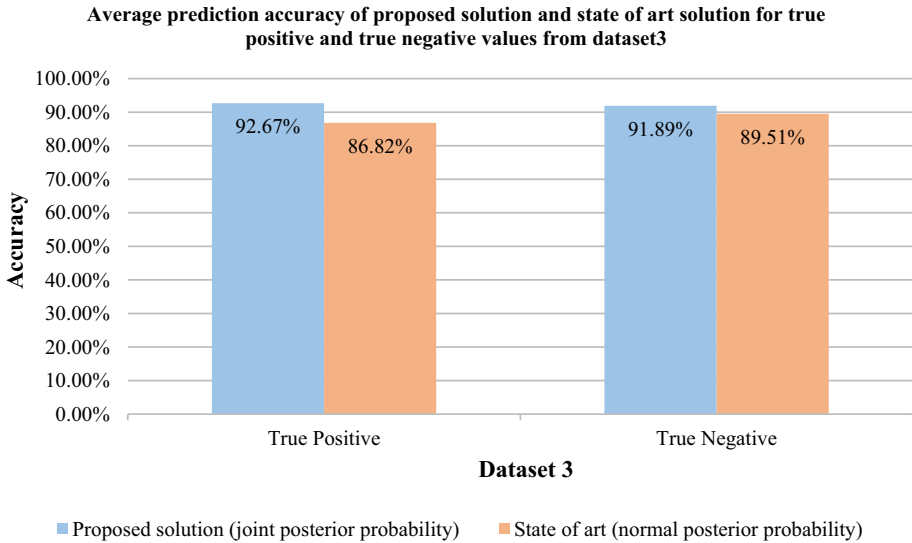


Fig. 12 Average prediction accuracy calculated for true positive and true negative results (in percentage) from Dataset 3. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average accuracy of true positive. **b** Second two lines show average accuracy of true negative

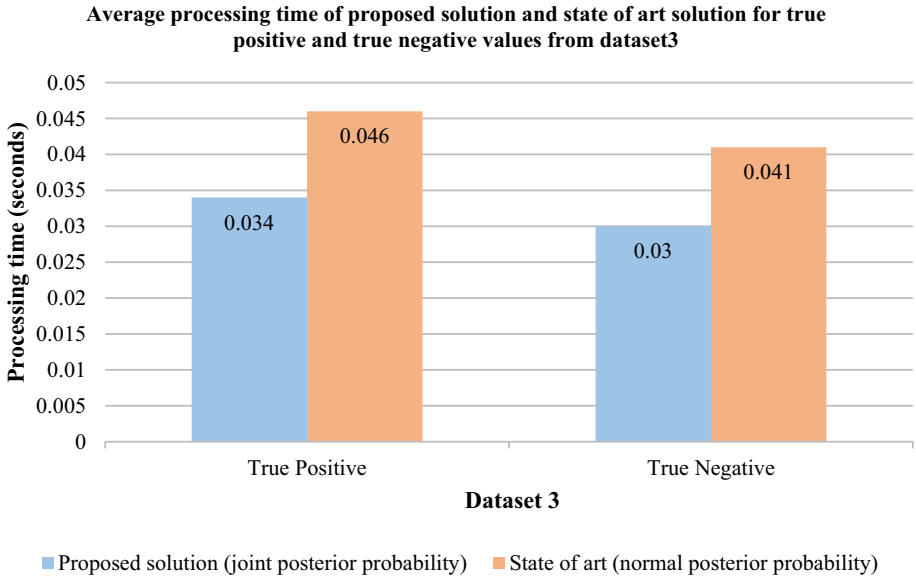


Fig. 13 Average processing time calculated for true positive and true negative results (in seconds) from Dataset 3. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average processing time of true positive. **b** Second two lines show average processing time of true negative

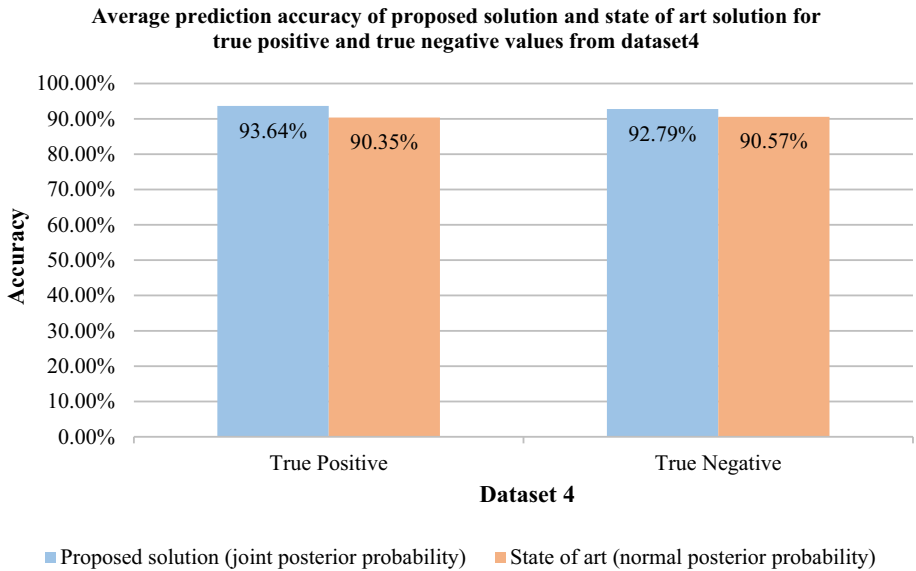


Fig. 14 Average prediction accuracy calculated for true positive and true negative results (in percentage) from Dataset 4. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average accuracy of true positive. **b** Second two lines show average accuracy of true negative

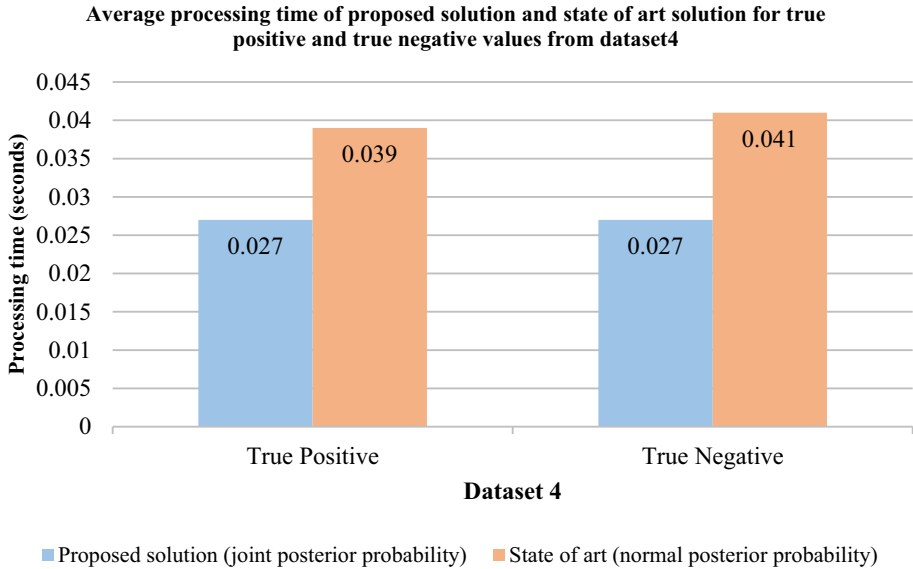


Fig. 15 Average processing time calculated for true positive and true negative results (in seconds) from Dataset 4. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average processing time of true positive. **b** Second two lines show average processing time of true negative

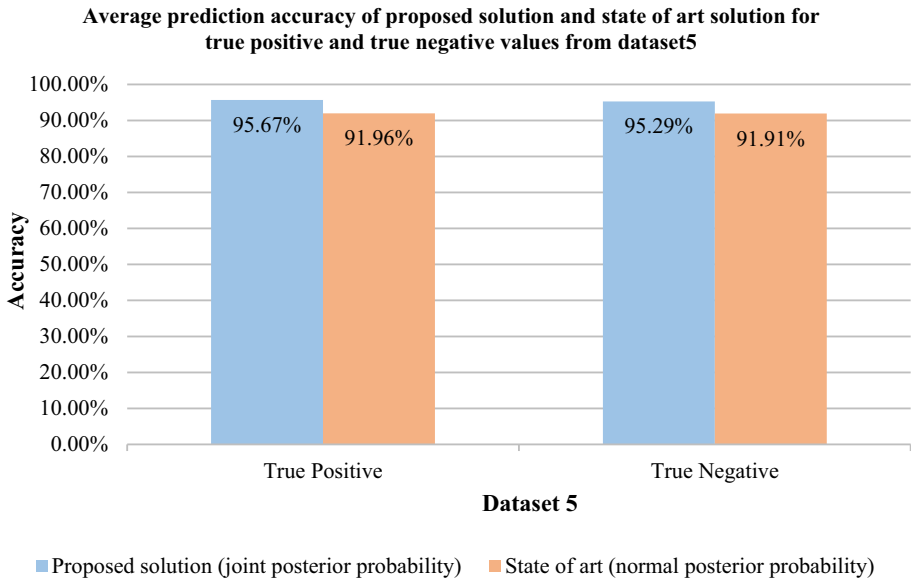


Fig. 16 Average prediction accuracy calculated for true positive and true negative results (in percentage) from Dataset 5. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average accuracy of true positive. **b** Second two lines show average accuracy of true negative

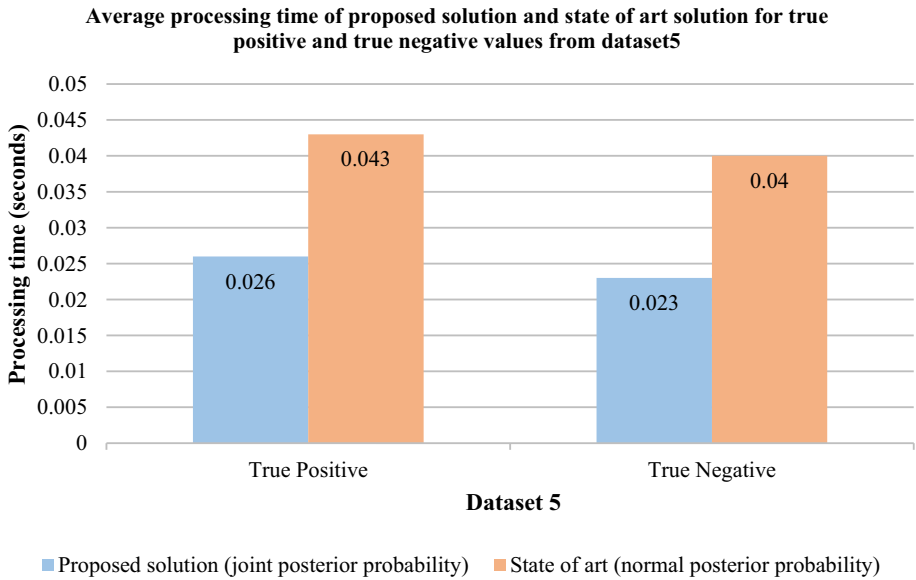


Fig. 17 Average processing time calculated for true positive and true negative results (in seconds) from Dataset 5. Blue lines show the proposed solution values. Orange lines show the state of art solution [2] values. **a** First two lines show average processing time of true positive. **b** Second two lines show average processing time of true negative

Table 5 After training and validation stages, the classification accuracy and processing time results of the proposed solution and the state of art solution for the five datasets

Dataset	Scenario	Stage	Proposed solution		State of art solution	
			Classification accuracy (%)	Processing time (epochs)	Classification accuracy (%)	Processing time (epochs)
Dataset1	1, 2, 3	Training	96	13	91	17
		Validation	88	7	81	10
Dataset2	1, 2, 3	Training	94	9	89	14
		Validation	83	5	80	10
Dataset3	1, 2, 3	Training	93	10	89	15
		Validation	86	4	79	9
Dataset4	1, 2, 3	Training	95	6	90	11
		Validation	87	2	83	9
Dataset5	1, 2, 3	Training	94	11	91	16
		Validation	81	5	83	12

Accuracy performance of Dataset 2 is shown in Fig. 5. Results compare the values of state of art [2] and the proposed solution during validation phase. When Fig. 5 examined, it can be observed that the proposed solution offers 4% more accurate classification accuracy compared to state of art solution. After training and validation stages, the results of all dataset classification accuracy and processing time can be seen in Table 5.

Table 6 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true positive values from Dataset 1

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	90.32	0.036	85.21	0.042
2. Duqu	88.25	0.046	81.54	0.053
3. Kazuar	82.12	0.044	70.12	0.048
4. Lazarus Group	73.01	0.041	60.24	0.059
5. Machete	86.87	0.051	74.87	0.069
6. NetTraveler	88.36	0.049	82.56	0.053
7. PowerDuke	90.89	0.043	86.52	0.048
8. Remexi	91.05	0.039	88.45	0.044
9. Soundbite	89.54	0.038	83.36	0.043
10. Winerack	89.74	0.044	86.32	0.049

Data report and bar graphs were used to compare the proposed solution results with the state of art system [2] in order to generate the presented tables and graphs. Results are based on three scenarios that are applied over the datasets. According to small, medium and large datasets, different results were obtained, and these results were used for comparison.

Results of different scenarios are evaluated according to the training and validation stages. Results of each dataset provide the accuracy and the processing time. To elaborate this; the accuracy values are determined by the ratio of correctly classified TTP samples to the total number of TTPs in the datasets [24]. The processing time values are calculated according to the number of predicted threats models that are required to reach reliable prediction level. The test dataset is 20% of the samples covered by the five datasets (Dataset1, Dataset2, Dataset3, Dataset4, and Dataset5).

Figure 6 shows the results of average accuracy that is calculated by the average of results obtained after the training phase and validation phase of respective five datasets. Figure 7 illustrates the average processing time, which is calculated by the average of results obtained after the training phase and validation phase of respective five datasets.

In the test phase, the average number of detected threats of each dataset was analysed based on three different scenarios. Results are presented in Tables 6, 7, 8, 9, 10, 11, 12, 13, 14 and 15. The prediction accuracy and the processing time values were obtained for TTPs based on the number of records in the datasets. While the probability of correct relation of TTPs is based on the prediction accuracy measurement.

The duration of classification process of TTPs is taken into consideration in the processing time measurement. As explained previously, results were analysed in two positions and the classification stage of belief network is the stage where results were obtained. With the help of Eq. (6), the proposed solution improves the prediction accuracy by enhancing threat posterior probability values. At the same time, according to Eq. (8) calculations, threat risk assessment is performed and the required processing time values are decreased for the prediction probability. The proposed system helps to identify the threat artifacts against most likely attack scenarios suggesting that real-time security analyses uses lowest cost and most likely mechanisms.

Table 7 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true negative values from Dataset 1

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	93.42	0.039	88.74	0.044
2. Duqu	91.21	0.044	84.55	0.054
3. Kazuar	85.45	0.049	73.48	0.043
4. Lazarus Group	76.69	0.041	63.12	0.055
5. Machete	89.47	0.053	77.33	0.067
6. NetTraveler	92.66	0.047	89.69	0.051
7. PowerDuke	90.79	0.045	86.47	0.049
8. Remexi	86.41	0.043	74.71	0.048
9. Soundbite	78.24	0.039	68.54	0.045
10. Winerack	89.36	0.042	76.98	0.046

Table 8 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true positive values from Dataset 2

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	91.42	0.034	87.26	0.043
2. Duqu	89.45	0.045	83.15	0.051
3. Kazuar	82.92	0.045	73.17	0.044
4. Lazarus Group	75.56	0.042	68.28	0.053
5. Machete	87.63	0.051	78.92	0.064
6. NetTraveler	87.82	0.049	83.04	0.049
7. PowerDuke	91.32	0.039	87.82	0.051
8. Remexi	92.43	0.042	89.08	0.046
9. Soundbite	88.97	0.039	84.81	0.041
10. Winerack	90.02	0.042	88.37	0.045

When results are evaluated, it is observed that the proposed solution model improves the prediction accuracy and the processing time values compared to the state of art model based on TTP classification. The proposed solution offers an average classification accuracy of 96% with Naive Bayes posterior probability and modified prior class probability using joint distribution functions. This result is 4% higher than the results of the state of art model. Moreover, the proposed solution achieves an average processing time of 0.028 s with the help of risk assessment for maximum support of set of the detected TTPs function. This value is 0.015 s less than the state of art solution.

Accuracy of datasets for each TTP was evaluated using the predict function of Python 3.6.9 Keras library. In this step, true positive and true negative were used to calculate the accuracy of correctly retrieved documents. In order to calculate the processing time values,

Table 9 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true negative values from Dataset 2

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	94.23	0.037	89.65	0.045
2. Duqu	91.78	0.041	83.14	0.048
3. Kazuar	86.01	0.046	77.24	0.046
4. Lazarus Group	77.32	0.042	67.18	0.045
5. Machete	89.49	0.049	81.42	0.057
6 .NetTraveler	91.97	0.048	88.01	0.051
7.PowerDuke	91.13	0.043	87.13	0.048
8. Remexi	85.14	0.042	76.71	0.047
9. Soundbite	79.86	0.041	69.98	0.045
10. Winerack	89.92	0.047	79.78	0.048

Table 10 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true positive values from Dataset 3

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	95.14	0.033	89.21	0.044
2. Duqu	94.18	0.022	93.74	0.050
3. Kazuar	94.74	0.046	76.22	0.045
4. Lazarus Group	88.54	0.041	88.16	0.050
5. Machete	88.41	0.029	79.77	0.052
6. NefTraveler	89.25	0.027	83.21	0.046
7. PowerDuke	96.13	0.039	89.15	0.048
8. Remexi	96.03	0.043	91.16	0.042
9. Soundbite	89.99	0.038	88.24	0.042
10. Winerack	94.24	0.029	89.33	0.041

Python 3.6.9 functions were used. Start time and end time intervals are determined with *Now method*. Moreover, the average accuracy and average time values were calculated with the help of Microsoft Excel average function. The improvements in the accuracy and the processing time values were investigated for the proposed solution against the state of art algorithms. In [2] the accuracy is calculated using Eq. (10):

$$\text{Accuracy} = \frac{\text{True positive}}{\text{True positive} + \text{True negative}} \quad (10)$$

where True positive: correctly retrieved TTPs from datasets dictionary. True negative: correctly dropped ttps from datasets dictionary.

In summary, using modified prior class probability threat support function (TFS), as the activation function in cyber-threat prediction algorithm, effectively avoids dependency prob-

Table 11 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true negative values from Dataset 3

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	95.21	0.033	89.36	0.040
2. Duqu	93.74	0.020	88.25	0.041
3. Kazuar	88.20	0.041	89.32	0.041
4. Lazarus Group	88.24	0.032	92.26	0.038
5. Machete	93.11	0.021	86.14	0.045
6. NetTraveler	93.97	0.028	91.37	0.041
7. PowerDuke	92.89	0.038	89.57	0.047
8. Remexi	87.18	0.022	89.21	0.039
9. Soundbite	89.97	0.039	92.88	0.038
10. Winerack	96.42	0.029	86.72	0.041

Table 12 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true positive values from Dataset 4

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	96.84	0.023	92.15	0.041
2. Duqu	94.22	0.032	89.14	0.035
3. Kazuar	89.16	0.036	92.21	0.041
4. Lazarus Group	91.14	0.024	89.24	0.042
5. Machete	89.47	0.029	92.15	0.039
6. NetTraveler	95.12	0.024	86.18	0.040
7. PowerDuke	96.87	0.031	88.10	0.039
8. Remexi	96.99	0.031	89.29	0.038
9. Soundbite	92.36	0.022	95.11	0.038
10. Winerack	94.25	0.023	89.92	0.040

lem of TTPs in the proposed solution model. TSF defines the best candidate threat prediction set with the maximum probability value. Bayesian probabilistic graphical model that is based on joint distribution calculates posterior probability. Therefore, this increases the probability accuracy as the graphical model of the Bayesian probabilistic effectively finds the best threat classification probability. Risk assessment function is another new feature of the proposed solution model. This function is used to identify the most relevant threats in the threat set, therefore, increases the accuracy of the probability function and reduces the processing time for threat prediction. As a result, it can be stated that the proposed solution provides increased accuracy and decreased processing time in cyber-threat prediction.

Various techniques have been used to detect and predict cyber-attacks. The most important limitation of these techniques has always been the attack prediction accuracy and the

Table 13 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true negative values from Dataset 4

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	96.41	0.030	91.12	0.044
2. Duqu	93.23	0.029	89.25	0.040
3. Kazuar	92.16	0.021	88.38	0.039
4. Lazarus Group	95.15	0.026	90.01	0.039
5. Machete	88.85	0.022	92.17	0.044
6. NetTraveler	92.14	0.027	87.39	0.042
7. PowerDuke	96.92	0.033	95.87	0.044
8. Remexi	89.58	0.032	87.91	0.038
9. Soundbite	89.47	0.031	93.80	0.041
10. Winerack	94.01	0.024	89.77	0.043

Table 14 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true positive values from Dataset 5

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction Accuracy (%)	Processing time (s)
1. Catchamas	96.15	0.023	93.54	0.046
2. Duqu	96.04	0.031	89.17	0.049
3. Kazuar	95.92	0.032	93.24	0.044
4. Lazarus Group	96.41	0.031	90.18	0.048
5. Machete	94.24	0.021	94.04	0.040
6. NetTraveler	94.18	0.025	95.19	0.038
7. PowerDuke	95.17	0.022	94.03	0.039
8. Remexi	95.12	0.027	90.10	0.041
9. Soundbite	96.95	0.030	89.02	0.044
10. Winerack	96.47	0.021	91.07	0.041

processing time during identification of attacks. The proposed solution solves the limitations encountered in the state of art model, achieving 96% prediction accuracy and achieving a 4% enhancement to the state of art solution which has 92% accuracy. At the same time, the proposed solution is superior to the state of art with processing time values. The proposed solution improves the state of art solution with an average processing time of 0.028 s against current processing time of 0.043 s. Threat support function is used to solve TTPs dependency problem, which is used as an activation function, and risk assessment function that improves processing time values were effective in obtaining improved results in different dataset scenarios. The main comparison between the proposed solution and the state of art is discussed in Table 16.

Table 15 Prediction accuracy–processing time results are given for the proposed solution and the state of art solution [2]. Results obtained for true negative values from Dataset 5

Sample TTP	Proposed solution		State of art solution	
	Prediction accuracy (%)	Processing time (s)	Prediction accuracy (%)	Processing time (s)
1. Catchamas	96.47	0.021	91.20	0.044
2. Duqu	96.12	0.024	95.13	0.041
3. Kazuar	92.17	0.031	89.01	0.037
4. Lazarus Group	97.04	0.022	91.29	0.039
5. Machete	93.35	0.022	95.72	0.041
6. NetTraveler	95.97	0.027	89.04	0.040
7. PowerDuke	96.72	0.023	90.21	0.042
8. Remexi	95.19	0.021	93.09	0.039
9. Soundbite	92.98	0.020	95.27	0.040
10. Winerack	96.87	0.021	89.17	0.041

5 Conclusion and Future Work

Methods and results that are presented with the proposed solution show that security incidents can be matched with cyber threat tactics in cyber threat intelligence. Machine learning can be used to artificially-link these mappings using specific possibilities and algorithms. In this context, it is worth noting that the prediction accuracy and the processing time are still limited. This study worked on the improvement of these two limitations. The proposed solution was inspired by the study that developed second best solution [7], and new features were developed such as modified version of Naive Bayes posterior probability and modified prior class probability. These functions increase the probability accuracy due to associated threats consideration compared to posterior Naive Bayes probability based on normalized conditional probability. Moreover, a new feature for risk management framework has been developed that allows the improvement in processing time limitation by using third best solution [3]. With posterior probability threat, risk assessment approach identifies the most relevant threats (using threat impact) in the cyber-threat set with increased accuracy of probability function and reduced time for threat prediction. Therefore, the proposed solution improves the average prediction accuracy by 4% and reduces the average processing time by 0.015 s. In future, in order to enable the developed model to be used in wider domains, multiple class datasets will be provided during testing and training stages of machine learning. In addition to that, studies will be carried out for mitigation integrations and automation of threat incidents detected in cyber threat intelligence. In this regard, development methods will be used to improve the threat classification performance and feature extraction.

Table 16 Comparison table between the proposed solution and the state of art solution

	Proposed solution	State of art solution
Name of the solution	Enhanced Naïve Bayes posterior probability (ENBPP)	A novel machine learning based framework
Applied area	Attack patterns based cyber threats	Attack patterns based cyber threats
Prediction accuracy	Prediction of attack pattern accuracy average is 96%	Prediction of attack pattern accuracy average is 92%
Detection time	Predict threat incidents for detected TTPs in an average time of 0.028 s	Predict threat incidents for detected TTPs in an average time of 0.043 s
Proposed equation	<p>Modified version of Naïve Bayes posterior probability</p> $M\mu(t_i, T_{TTP_i} ttp_i) = \left(\frac{\omega(ttp_i t_i)}{\sum_{t_i \in T_{TTP_i}} \omega(ttp_i t_i)} \right) Mp(t_i)$ <p>Enhanced Naïve Bayes posterior probability (ENBPP)</p> $ES(t_i) = \frac{\sum_{TTP_i \in TTP_{Di}} M\mu(t_i, T_{TTP_i} ttp_i)}{\sum_{TTP_i \in TTP_{Ti}} M\mu(t_i, T_{TTP_i} ttp_i)} + MR_{Ti}$	<p>Posterior Naïve Bayes probability</p> $\mu(t_i ttp_i) = \frac{\omega(ttp_i t_i) p(t_i)}{\sum_{t_i \in T_{TTP_i}} \omega(ttp_i t_i) p(t_i)}$
Used datasets and samples	Processing time and classification accuracy results were obtained by using five different datasets containing 328,814 (Dataset1: 36,222 samples, Dataset2: 136,814 samples, Dataset3: 25,038 samples, Dataset4: 47,622 samples, Dataset5: 83,118 samples) threat samples.	State of art solution created three different datasets, and used 133,450 (Dataset1: 45,019 samples, Dataset2: 23,574 samples, Dataset3: 64,857 samples) threat samples.
Contribution 1	Activation function is modified threat support function based on Bayesian probabilistic graphical model, which increases probability accuracy.	Threat support function used as an activation function in the system faces dependency problems of TTPs. This affects ability to recognize attacks and reduces overall threat prediction reliability.
Contribution 2	Risk management framework during threat prediction stage considers treat probability after risk assessment to generate maximum support results of detected TTPs. This effectively prevents threshold mistakes in prediction sets, improving processing time and enhances reliable threat prediction values.	State of art does not provide a solution to deal with prediction reliability problems.

References

1. Qamar S, Anwar Z, Rahman MA, Al-Shaer E, Chu BT (2017) Data-driven analytics for cyber-threat intelligence and information sharing. *Comput Secur* 67:35–58
2. Noor U, Anwar Z, Malik AW, Khan S, Saleem S (2019) A machine learning framework for investigating data breaches based on semantic analysis of adversary’s attack patterns in threat intelligence repositories. *Future Gener Comput Syst* 9:467–487. <https://doi.org/10.1016/j.future.2019.01.022>

3. Riesco R, Villagra VA (2019) Leveraging cyber threat intelligence for a dynamic risk framework. *Int J Inf Secur* 18:715–739. <https://doi.org/10.1007/s10207-019-00433-2>
4. Xiao Y, Xing C, Zhang T, Zhao Z (2019) An intrusion detection model based on feature reduction and convolutional neural networks. *IEEE Access* 7:42210–42219. <https://doi.org/10.1109/access.2019.2904620>
5. Lee J, Kim J, Lim I, Han K (2019) Cyber threat detection based on artificial neural networks using event profiles. *IEEE Access* 7:165607–165626. <https://doi.org/10.1109/access.2019.2953095>
6. Husak M, Komarkova J, Bou-Harb E, Celeda P (2019) Survey of attack projection, prediction, and forecasting in cyber security. *IEEE Commun Surv Tutor* 21(1):640–660. <https://doi.org/10.1109/comst.2018.2871866>
7. Sun P, Li J, Bhuiyan ZA, Wang L, Li B (2019) Modelling and clustering attacker activities through machine learning techniques. *Inf Sci* 479:456–471. <https://doi.org/10.1016/j.ins.2018.04.065>
8. Caminero G, Martin ML, Carro B (2019) Adversarial environment reinforcement learning algorithm for intrusion detection. *Comput Netw* 159:96–109. <https://doi.org/10.1016/j.comnet.2019.05.013>
9. Noor U, Anwar Z, Amjad T, Kwang K, Choo R (2019) A machine learning-based FinTech cyber threat attribution framework using high-level indicators of compromise. *Future Gener Comput Syst* 96:227–242. <https://doi.org/10.1016/j.future.2019.02.013>
10. Kim D, Kim HK (2019) Automated dataset generation system for collaborative research of cyber threat analysis. *Secur Commun Netw* 2019:1–10. <https://doi.org/10.1155/2019/6268476>
11. Franssen F, Smulders A, Kerkdijk R (2015) Cyber security information exchange to gain insight into the effects of cyber threats and incidents. *Elektrotech Inf Technol* 132(2):106–112
12. Du M, Li F, Zheng G, Srikumar V (2017) DeepLog: anomaly detection and diagnosis from system logs through deep learning. *Proc ACM CCS* 17:1285–1298
13. Subroto A, Apriyana A (2019) Cyber risk prediction through social media big data analytics and statistical machine learning. *J Big Data* 6:1–19. <https://doi.org/10.1186/s40537-019-0216-1>
14. Kaja N, Shaout A, Ma D (2019) An intelligent intrusion detection system. *Appl Intell* 49:3235–3247. <https://doi.org/10.1007/s10489-019-01436-1>
15. Black P, Gondal I, Layton R (2018) A survey of similarities in banking malware behaviours. *Comput Secur* 77:756–772. <https://doi.org/10.1016/j.cose.2017.09.013>
16. Li G, Shen Y, Zhao P, Lu X, Liu J, Liu Y, Hoi SCH (2019) Detecting cyber-attacks in industrial control systems using online learning algorithms. *Neurocomputing* 364:338–348
17. Durkota K, Lisya V, Bosanskya B, Kieikintveld C, Pechoucek M (2019) Hardening networks against strategic attackers using attack graph games. *Comput Secur* 87:1–25. <https://doi.org/10.1016/j.cose.2019.101578>
18. Gu J, Wang L, Wang H, Wang S (2019) A novel approach to intrusion detection using SVM ensemble with feature augmentation. *Comput Secur* 86:53–62
19. Gao X, Shan C, Hu C, Niu Z, Liu Z (2019) An adaptive ensemble machine learning model for intrusion detection. *IEEE Access* 7:82512–82521. <https://doi.org/10.1109/access.2019.2923640>
20. Bahtiyar S, Yaman YB, Altinigne CY (2019) A multi-dimensional machine learning approach to predict advanced malware. *Comput Netw* 160:118–129. <https://doi.org/10.1016/j.comnet.2019.06.015>
21. Li Y, Xiong K, Chin T, Hu C (2019) A machine learning framework for domain generation algorithm-based malware detection. *IEEE Access* 7:32765–32782. <https://doi.org/10.1109/access.2019.2891588>
22. Antunes M, Gomes D, Aguiar RL (2018) Towards IoT data classification through semantic features. *Future Gener Comput Syst* 86:792–798. <https://doi.org/10.1016/j.future.2017.11.045>
23. Huda S, Abawajy J, Alazab M, Abdollalihan M, Islam R, Yearwood J (2016) Hybrids of support vector machine wrapper and filter based framework for malware detection. *Future Gener Comput Syst* 55:376–390. <https://doi.org/10.1016/j.future.2014.06.001>
24. Qublai K, Mirza A, Awan I, Younas M (2018) CloudIntell: an intelligent malware detection system. *Future Gener Comput Syst* 86:1042–1053. <https://doi.org/10.1016/j.future.2017.07.016>