Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2019) 27: 392 – 406
© TÜBİTAK
doi:10.3906/elk-1804-55

Improving anomaly detection in BGP time-series data by new guide features and moderated feature selection algorithm

Mahmoud HASHEM*, **Ahmed BASHANDY**, **Samir SHAHEEN**
Department of Computer Engineering, Faculty of Engineering, Cairo University, Giza, Egypt

Received: 07.04.2018 • **Accepted/Published Online:** 06.11.2018 • **Final Version:** 22.01.2019

Abstract: The Internet infrastructure relies on the Border Gateway Protocol (BGP) to provide essential routing information where abnormal routing behavior impairs global Internet connectivity and stability. Hence, employing anomaly detection algorithms is important for improving the performance of BGP routing protocol. In this paper, we propose two algorithms; the first is the guide feature generator (GFG), which generates guide features from traditional features in BGP time-series data using moving regression in combination with smoothed moving average. The second is a modified random forest feature selection algorithm which is employed to automatically select the most dominant features (ASMDF). Our mechanism shows that the detected anomalies are more realistic and the selected features are generally consistent across time series. Experimental evaluations using multiple machine learning models reveal that the proposed algorithms achieve up to 32.36% improvement in accuracy rate, up to 35.44% reduction in false negative rate, and up to 43.99% reduction in false positive rate compared to not using these algorithms. Moreover, the ASMDF option increases the feature selection speed more than 3 times compared to most existing feature selection algorithms.

Key words: Detecting anomalies, feature selection, machine learning, time series

1. Introduction

The Border Gateway Protocol (BGP) is used to exchange routing information between border routers in a network comprising many autonomous systems. BGP generates four types of messages: open, update, keep alive, and notification, which are exchanged among BGP peers based on a set of metrics such as the nearest next-hop router, the shortest autonomous system path, and routing policies. While messages are exchanged among peers, BGP anomalies may be triggered by a variety of events such as router misconfigurations, session resets, and link failures. The simplest definition of BGP anomalies is any update that does not reflect a change in the underlying BGP network or routing policy. These anomalies degrade network performance and efficiency.

Many techniques have been employed to detect BGP anomalies [1]. Unfortunately, these existing anomaly detection methods select the traffic features of the present to make the decision regardless of the time series of the traffic data, where time-series analysis can bring extra important information in identifying state changes. Considering this limitation, we proposed a time-series unit that consists of three stages: feature extraction, feature selection, and machine learning stage. The main aim is speeding up and improving the performance of the anomaly detection process in the case of real-time detection. It can also be used if we have a huge number of update messages stored and want to process these messages as time-series data.

For the first stage, we propose the guide feature generator (GFG) algorithm. The main functions of

*Correspondence: mahmhashem@gmail.com

this algorithm are to receive series of update messages, extract traditional and new guide features from these messages, mark periods that represent abnormal activities, and output successive vector sets of extracted features. To mark abnormal periods, the GFG uses our proposed equation that employs moving regression in combination with smoothed moving average, which is commonly used with time-series data. As a result, it is easy to select the vector sets that contain abnormal activities and test them in the third machine learning stage and neglect the normal vector sets. The new guide features generated by the GFG algorithm improve the classification results.

In the second stage, the small groups of successive vector sets produced by the first stage are passed to the proposed algorithm to automatically select the most dominant features (ASMDF) to select the most dominant features for each abnormal vector set only, where each vector set has different dominant features. We propose this modified feature selection algorithm for three reasons: to adapt with incoming series data, to directly pass each abnormal vector set with its dominant features to the third stage, and to stop executing the third stage for unmarked normal vector sets where the third stage is any selected machine learning technique, which receives each vector set with its dominant features from the second stage, for performing the classification process. This leads to decreasing processing time and computations.

Moreover, the ASMDF algorithm automatically selects the most dominant subset of features by employing the random forest mechanism [2], and we suggest two improvements to speed up its computations. First is to generate a small group of successive vector sets by the GFG algorithm, and second is to use orientation rather than magnitude (vector set values) by applying the cosine similarity equation [3] on the generated vector set to produce a smaller cosine similarity matrix that generates fewer random forest trees compared to its corresponding vector set. For this purpose, we add the cosine similarity equation as underlying the ASMDF algorithm.

In summary, the contributions of this work are as follows:

- **In stage 1:** The proposed GFG algorithm handles time-series data in an efficient way using our proposed time-series equation and marks the abnormal vector sets. Moreover, using the guide features that are generated by the GFG raises the machine learning classification accuracy improvement rate up to 32.36%, improvement in decreasing the false negative rate up to 35.44%, and improvement in decreasing the false positive rate up to 43.99% during the testing phase.

- **In stage 2:** The proposed ASMDF algorithm is modified to adapt to the time-series data. Moreover, it is used to process the abnormal vector sets only and passes each vector set with its dominant features to the third stage. This leads to decreasing processing time and computations in the third machine learning stage by testing only the abnormal vector sets. Besides, using the cosine similarity as underlying the ASMDF increases the feature selection speed more than 3 times compared to most existing feature selection algorithms.

This paper is structured as follows. In Section 2, we present a brief literature review related to our work. Section 3 describes our methodology for gathering BGP update messages and discusses the proposed scenario. Section 4 discusses the proposed algorithms. Section 5 presents the simulation results using supervised machine learning techniques. Section 6 concludes the paper and future work.

2. Related work

Many techniques have been proposed to identify anomalies by analyzing traffic patterns. One of the early and common methods is developing traffic behavior models based on statistical techniques [4, 5], where the anomalies are identified as correlated abrupt changes occurring in the underlying distribution. However, the disadvantage is that it is difficult to estimate the dimension distributions with all possible cases. Also, clustering techniques [6] were proposed to classify all regular traffic data points belonging to one cluster while anomalous data points may belong to multiple clusters. The main disadvantage of the clustering techniques is that they are optimized to find the regular traffic rather than the anomalous traffic, which is usually the goal of the detection techniques. Another widely used approach is the rule-based technique [7], which builds classifiers based on a set of rules. The drawback is that it requires a priori knowledge and a high degree of computations. Many machine learning techniques have been employed to build traffic classification models [8, 9] for both unsupervised and supervised machine learning models to detect anomalies. Although neural networks have the ability to detect the complex relationship among features, they have many drawbacks such as high computational complexity and high probability of overfitting. Support vector machine (SVM) techniques detect the anomaly patterns in data using nonlinear classification functions and classify each data point based on its value obtained by the classifier function. SVMs build a classification model that maximizes the margin between the data points that belong to each class. Several variants of SVM detection techniques are introduced and evaluated [10], but they have high computational complexity because of the quadratic optimization problem that needs to be solved. Finally, Bayesian networks (BN) techniques are used in many real-time classification systems because of their low time complexity. BNs rely on two assumptions: the features are conditionally independent given a target class and the posterior probability is the classification criterion between any two data points. Several anomaly detection schemes have implemented variants of BNs [11, 12].

Most models mentioned here are not designed for sequence classification and are not suitable for anomaly detection in time series, where they only treat the input instances independently without considering the sequenced nature of traffic data. In reality, traffic data are multivariant time series and the anomaly patterns vary gradually with the temporal information. For this reason, we propose our work.

3. BGP data and methodology

The Réseaux IP Européens Network Coordination Centre (RIPE NCC) collects and stores Internet routing data through the Routing Information Service project (www.ripe.net). BGP update messages are collected by remote route collectors (RRCs) and stored in Multithreaded Routing Toolkit (MRT) binary format (available: RFC 6397). BGP update messages can originate from multiple monitoring points (collectors), where RIPE NCC operates several collectors and each monitoring point peers with multiple operational BGP routers at various ISPs. We converted multiple BGP update messages from MRT into ASCII format by using the bgpdump library on the Linux platform. bgpdump is a C library maintained by RIPE NCC that is used to analyze dump files produced by MRT. We collected these BGP update messages during the time periods that experienced anomalies. These BGP update messages were collected specifically from two collectors (RRC02 and RRC04) up to 2017 in 10 groups. These groups were collected with an average duration of 3 days per group. During these periods many changes in BGP behavior, such as BGP hijacking [13], worm attack [14], and link failures [15], are detected. The collected groups contain the following: the peak day of an anomaly, one day prior, and one day after the anomaly. This sampling yields a huge number (22×10^6) of collected BGP update messages.

3.1. The proposed scenario

For clarification, we designed our proposed algorithms to handle BGP update messages as time-series data in an attempt to speed up and enhance the process of detecting anomalies. In the first stage, the number of BGP update messages that are received in 1 min by the GFG algorithm is called a block, as shown in Figure 1.

While the GFG algorithm receives a series of blocks, it extracts features from these blocks and samples them every minute (row/min). The GFG algorithm produces two types of features: traditional features, which are extracted from BGP update messages, and guide features, which are generated from traditional features. There are 16 traditional and 16 guide features, as shown in Table 1. The second step of the GFG algorithm is to mark the anomalous periods by employing the guide features that are generated by applying moving average and moving regression to the corresponding traditional feature as explained in Section 4.1. The guide feature values (each value is one binary bit, "1" for abnormal and "0" for normal) in each vector set are used to quickly identify the periods that are likely to contain anomalies or imbalances. If these guide feature values assign 1 to a long period crossing multiple vector sets, this is an indicator of abnormal activities. Based on these abnormal periods, only a specific number of vectors that have abnormal values in guide feature(s) are used to perform the classification quickly and without consuming much time in processing all incoming time-series data. Moreover, one of the advantages of employing moving average and moving regression is to generate each guide feature from its traditional feature at the same time or step by step; this means that there is synchronization between producing both types of features. Finally, the GFG algorithm outputs small groups of successive vector sets, where each vector set is assembled in a random time less than 12 min by the GFG algorithm and has multiple rows for 32 features.

In the second stage, the proposed ASMDF algorithm, which is designed to handle time-series data, receives these vector sets as shown in Figure 1. The ASMDF algorithm checks if the vector set is marked by the GFG algorithm (containing abnormal values in guide features). For marked vector sets only, the ASMDF algorithm selects the most dominant features for each vector set where each vector set has its own different dominant features and directly passes the vector set with its dominant features to the third machine learning stage. The third stage is any selected machine learning technique, which receives each marked vector set with its dominant features from the second stage, for performing the classification process. This stage does not test the unmarked vector sets, which leads to decreased processing time and computations. The pseudocode for the time-series unit (3 stages) is presented in Table 2. This can be applied in the case of real-time detection and also if we have a huge number of update messages stored and want to process these messages as time-series data. For better clarification, Section 4.1 describes how the guide feature is generated from its corresponding traditional feature.

4. The proposed algorithms

In this section, we describe our proposed algorithms. These algorithms are used to handle different types of anomalies or abnormal network activities that appear in BGP update messages. The proposed algorithms consider the generated samples as a time series of BGP data traffic.

4.1. The GFG algorithm

The GFG algorithm receives a time series of update messages, handles it as a sequence of blocks, and generates vector sets that contain two types of features: traditional and guide, as described in Section 3.1. The GFG algorithm is not for feature selection but for improving detection, where any traditional feature can be used to generate a guide feature, thereby helping to quickly identify periods in time-series data that may represent abnormal activities. Each guide feature is generated by applying moving average and moving regression to its corresponding traditional feature and the output is in the form of individual binary bits. The result is that, in the vector set, each guide feature contains the values that are likely to represent anomalies or imbalances

Table 1. Features generated by the GFG algorithm.

Traditional features	Definition of traditional features	Guide features[c]
ANUM	Number of announcement update messages	g.ANUM
WNUM	Number of withdrawal update messages	g.WNUM
IGP	Number of BGP update messages generated by an interior gateway protocol	g.IGP
EGP	Number of BGP update messages generated by an exterior gateway protocol	g.EGP
INCOM	Number of incomplete update messages generated by unknown sources	g.INCOM
AIP	Number of announced NLRI[a] prefixes inside BGP update messages	g.AIP
WIP	Number of withdrawn NLRI[a] prefixes inside BGP update messages.	g.WIP
ASMIN	Minimum AS-PATH length	g.ASMIN
ASMAX	Maximum AS-PATH length	g.ASMAX
ASAVG	Average AS-PATH length	g.ASAVG
DUBA	Number of duplicate announcement BGP update messages	g.DUBA
DUBW	Number of duplicate withdrawal BGP update messages	g.DUBW
IMPAW	Number of update messages announced and then withdrawn for the same prefix	g.IMPAW
MXDST	Maximum edit distance	g.MXDST
AVDST	Average edit distance	g.AVDST
MSARR	Average message interarrival time	g.MSARR

[a] Network layer reachability information, [b] calculated from traditional, [c] binary bits.

in traffic (anomalies or imbalances as 1 and 0 otherwise). If these guide feature values are assigned 1 for long period crossing multiple vector sets, it is an indicator of anomalies or imbalances.

The algorithm behaves according to the following five steps for each block of the algorithm:

Step 1. Extracts the traditional features.

Step 2. Calculates the moving average [16] for each traditional feature to smooth out the feature values using Eq. (1).

$$V_t = \frac{1}{2K+1} \sum_{j=-K}^{K} d_{t+j} \tag{1}$$

Here, V_t is the moving average smoother, d_t is the time-series data, and K is the average block size. The moving average is commonly used with time-series data to smooth out short-term fluctuations and highlight longer-term trends or cycles.

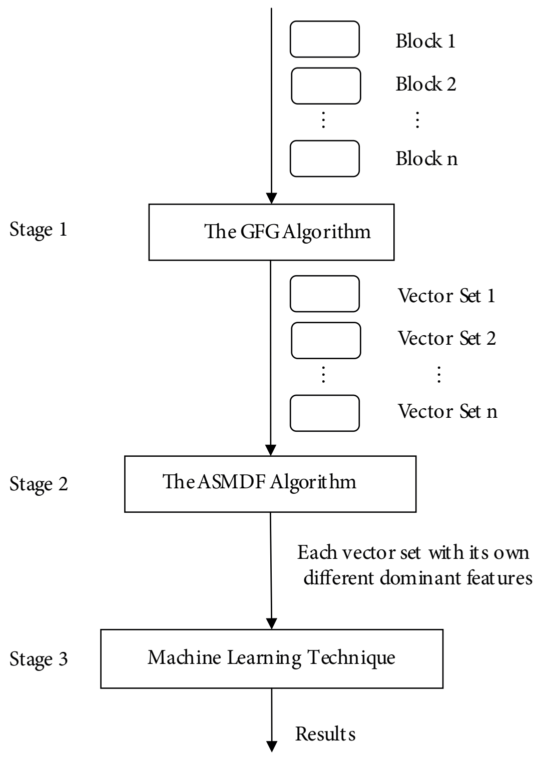

Figure 1. Time-series unit (3 stages).

Table 2. Time-series unit (3 stages) pseudocode.

```
Input: BGP block, number of update messages in 1 min
Output: Machine learning classification results
Stage 1 (The GFG algorithm):
  Set vector set assembler timer
  if new block received do
  Extract traditional and guide features
  Mark abnormal activity periods
  if vector set assembler timer reset then
  Generate new vector set and send it to Stage 2
  else wait a new block
  repeat with frequently input data
Stage 2 (The feature selection algorithm):
  if normal vector set received then wait new vector set
  else if abnormal vector set received do
  Select the most dominant features
  Send the vector set with its dominant features to Stage 3
  repeat with frequently input data
Stage 3 (Machine learning classification process)
```

Step 3. Calculates the moving regression for each traditional feature using Eq. (2).

$$y_t = x_t\beta + \varepsilon_t \tag{2}$$

Here, y_t is the dependent variable (also called the response variable), such as the g.ANUM feature; x_t is the independent variable (also called the predictor); ε_t is an unobserved random variable, known as the error or disturbance term; and β is the coefficient or slope parameter, which represents the slope of the straight line that the equation describes. Moving regression is derived from basic regression [17, 18], also known as rolling regression or recursive regression, which is often used in time-series analysis to assess the stability of the model parameters with respect to time and finding the best-fitting line through the points. The best-fitting line (also called the regression line) represents the average relationship between the response variable (y_t) and the predictor variable (x_t).

We assume that x_t is the initial probability of attack, which is calculated only once for each traditional feature during every block period, where x_t are output fractional values between 0 and 1, calculated using Eq. (3).

$$x_t = 1 - \left(\frac{Fopt_t}{Favg}\right) \tag{3}$$

Here, $Fopt_t$ is the optimal average traditional feature value and is calculated based on normal feature values experienced over a long period of time crossing multiple blocks, and Favg is the average value for that traditional feature during the block period. $Fopt_t$ is updated based on the latest guide feature values during normal traffic, i.e. when the guide feature values are 0. During anomalous periods, the Favg values are very large and x_t is close to 1, while in normal conditions Favg is approaching $Fopt_t$ and x_t is close to 0. y_t is the traditional feature such as ANUM, IGP, and AIP features. The outputs from Eq. (2) are stored as R_t, where R_t is the moving regression output values using Eq. (2).

The moving regression values are calculated for each traditional feature, and the calculations are repeated on the data sequence as long as data blocks are received by the GFG algorithm. The GFG algorithm produces constant moving regression values for each traditional feature during each block period because each traditional feature has its own constant initial probability of attack value (x_t). The initial probability of attack value (x_t) for any traditional feature changes only when moving from one block to the next block. In general, the regression line for any traditional feature is an output line that changes its constant value when moving from one block to another.

Step 4. Calculates the deviation rate DR_t for each block received by the GFG algorithm. Deviation rate DR_t is a novel attribute defined in Eq. (4).

$$DR_t = R_t + \sigma\left(V_t\right) \tag{4}$$

Here, R_t is the moving regression output values using Eq. (2), V_t is the moving average calculated using Eq. (1), and $\sigma\left(V_t\right)$ is the standard deviation of the moving average that is calculated only once using Eq. (5) during each block period.

$$\sigma\left(V_t\right) = \sqrt{\sum_{i=1}^{N} p_i\left(V_t - \mu\right)^2}, where \quad \mu = \sum_{i=1}^{N} p_i V_i \tag{5}$$

Here, μ is the mean, N is the number of values, p_i is the probability of V_i, and $\sigma\left(V_t\right)$ is the standard deviation of the moving average V_t.

Adding the standard deviation to the moving regression as in Eq. (4) causes the output line to be very close to the maximum y_t values without exceeding it. Clearly, the y_t values increase as the probability of attack increases while the number of messages exchanged among the routers increases dramatically and for periods that may be long.

Step 5. Finally, the intersection of the smoothed time-series data points (V_t) and the deviation rate output values (DR_t) over time may be very useful as an indicator of anomalies or imbalances in traffic. For example, see Figures 2 and 3.

One of the advantages of employing moving average and moving regression is that both the smoothed line and the deviation rate line are generated at the same time, which means there is synchronization in producing the two lines. We take advantage of the output of these intersections and employ it as a new guide feature, which may be very useful in detecting any defects or attacks experienced by the network. For this reason, the obtained results are recorded as binary values, where each intersection is represented by a value of 1 and otherwise 0 and stored as a new guide feature, such as the g.ANUM feature in Figure 4.

For a better understanding of this algorithm, suppose that a slammer attack appears in the time-series data where y_t is the number of announcement BGP update messages “ANUM”, which are increased during the slammer attack for a long period of time as shown in Figure 2. The intersections that were mentioned before can be shown in Figure 3; the smoothed feature line represents the smoothed ANUM feature values, which are calculated using Eq. (1), while the deviation rate line represents the sequence of deviation rate values, which are calculated using Eq. (4), and the high spikes that appear for a short period time do not belong to any attack but rather are due to open and closed BGP sessions. The output results are recorded as binary values, where each intersection is represented by a value of 1 and otherwise 0 and stored as a new feature called the g.ANUM as shown in Figure 4. The pseudocode for the GFG algorithm is presented in Table 3.

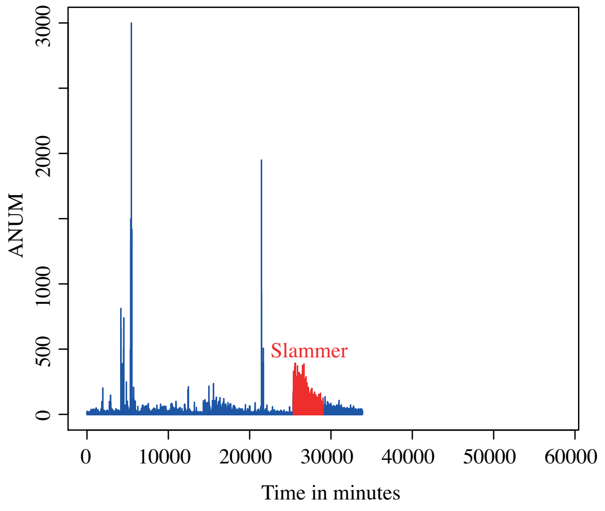

Figure 2. Slammer attack as appeared in ANUM feature.

In this paper, we did not focus on the differences between anomalies and other imbalance types; instead, we only focus on detecting anomalies by employing suitable machine learning techniques. For other imbalances types, we will employ different techniques as a future work.

4.2. The proposed feature selection algorithms

4.2.1. The ASMDF algorithm

Feature selection algorithms are categorized as filter, wrapper, and hybrid [19]. The proposed ASMDF algorithm is a modified feature selection algorithm, which has the advantage of dealing with time-series data and makes use of the effective random forest tree model. The random forest is a versatile machine learning technique

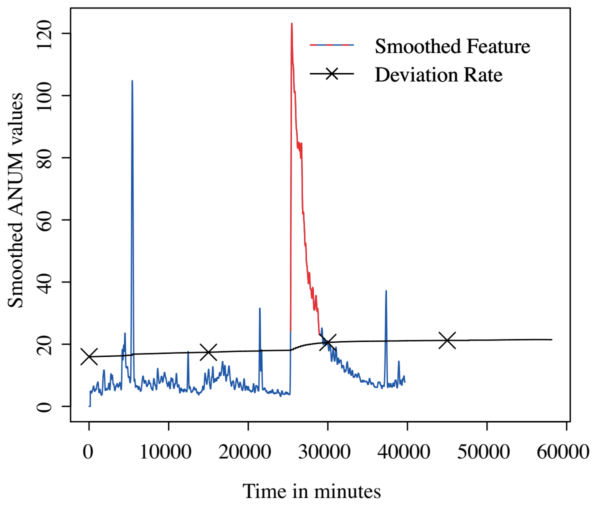

Figure 3. Smoothed ANUM feature and its deviation rate.

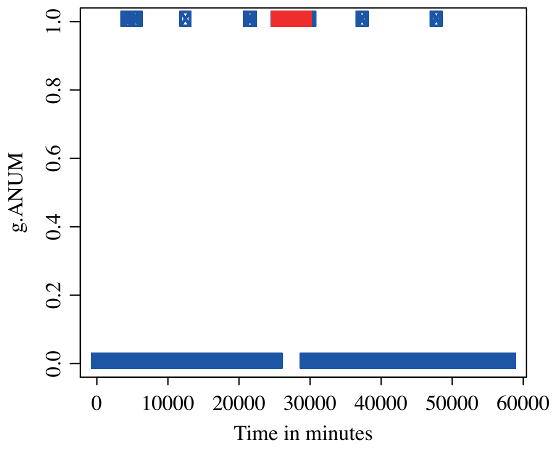

Figure 4. The g.ANUM feature (intersection points).

Table 3. The GFG pseudocode.

```
Input: Blocks of update messages
Output: Vector sets
Set block timer
Set vector set assembler timer
if block received begin
Extract traditional features
Smooth extracted features, calculating the moving average, $V_t$
Calculate the moving regression values for each feature, $R_t$
   if end of block begin
   Calculate the standard deviations $\sigma(V_t)$ for $V_t$
   Calculate the deviation rates, $DR_t = R_t + \sigma(V_t)$
   Measure $R_t$ and $V_t$, the “intersection results”
   Record the guide features’ binary results
   increment block counter and reset timer
   end
if vector set assembler timer reset then
Generate new vector set and send it to Stage 2
end
repeat with frequently input data
```

capable of performing both regression and classification tasks but may be very slow, especially when using a large number of trees to predict final results. To speed up the feature selection computations, we suggest two improvements: the first is to generate a small group of successive vector sets by the GFG algorithm and the second is to use orientation rather than magnitude (vector set values) by applying the cosine similarity equation to the vector set to produce a smaller cosine similarity matrix that generates fewer random forest trees compared to the corresponding vector set. For this purpose, we add the cosine similarity equation as underlying the ASMDF algorithm where the cosine similarity is a measure of similarity between two nonzero features of an inner product space that measures the cosine of the angle between them. It is thus a judgment of orientation and not magnitude: two features with the same orientation have a cosine similarity of 1, two features at $90°$ have a similarity of 0, and two features diametrically opposed have a similarity of -1, independent of their magnitude. Given two features, A and B, the cosine similarity, C, is represented using a dot product and magnitude as in Eq. (6), where A_i and B_i are components of feature A and B, respectively, of length n.

$$C = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{6}$$

For each abnormal marked vector set, the ASMDF algorithm calculates the cosine similarity matrix between all traditional features and returns a similarity matrix of cosine values. The similarity matrix values are orientations and not a series of individual values (magnitude), which results in the size of the matrix being smaller compared to using a series of individual values (vector set values). The ASMDF algorithm uses the cosine matrix to generate the random forest trees, which are fewer in number compared to the corresponding vector set. This process leads to generating a small number of trees during the random forest phase and therefore speeds up the features selection computations. Also, it may be better not to execute the algorithm continuously but rather intermittently, especially when suspicious activity is observed or by selecting vector sets at intervals that may be converged or diverged, to ensure that no long time-consumption in processing. The pseudocode is presented in Table 4. For each abnormal vector set, the dominant features are produced by using the cosine similarity and random forest trees. Then these dominant features are employed to obtain the best machine learning classification.

Table 4. The ASMDF pseudocode.

```
Input: Vector sets
Output: The dominant features
Initial vector set counter
if normal vector set received then wait new vector set
else if abnormal vector set received begin
  Calculate the cosine similarity matrix for the vector set
  if cosine matrix calculated begin
  Generate the random forest trees from the cosine matrix
  Output the dominant features
  Send the vector set with its dominant features to Stage 3
  end
end
repeat with frequently input data
```

4.2.2. The feature selection strategy

The ASMDF algorithm selects the dominant features only from the traditional features while every guide feature follows its corresponding dominant feature. Logically, the guide features are not actual data, but 0's and 1's that are considered the intersection points. It is like two tied layers, a primary layer (traditional) and secondary layer (guide). If the ASMDF algorithm selects two traditional features this means that the vector set has four dominant features, two traditional and two guide features.

5. Simulation results

The simulator (R language on Intel CPU 2.6 GHz, 64-bit 4GB RAM, running Linux system) was run several times for 7 days for generating average results, using both the traditional and the guide features. In this section, we focus on CodeRed-I, Slammer, and Nimda attacks as examples or samples and employ a variety of machine learning techniques to evaluate our proposed algorithms. We extracted periods from the update messages, which are collected as mentioned in Section 3, and contain these three types of attacks to build the training datasets as shown in Table 5. Moreover, the test instances for each dataset are presented in two ways: the total number of instances and the number of vector sets that are used to simulate the time-series data, where the vector sets are counted by the ASMDF algorithm. Also, the ASMDF algorithm selects different dominant features for each vector set, and every guide feature follows its corresponding dominant feature as explained in Section 4.2.2. The average block size is K = 100 for the moving average smoother. The simulation metrics are: Tsec, presents the training time in seconds; ACC, the ACCuracy for the testing phase; false negative rate (FN), the proportion of positive cases (attacks) incorrectly classified as negative (not attacks); and false positive rate (FP), the proportion of negative cases (not attacks) incorrectly classified as positive (attacks).

In the next subsections, we analyze the performance with and without the guide features generated by the proposed GFG algorithm. Moreover, we compare the ASMDF algorithm with existing feature selection algorithms to show the differences between their speeds in selecting the dominant features for the vector set, as seen in Section 5.4.

Table 5. Training and testing datasets.

Data set	Train instances	Test instances
Slammer-Normal	8723	10,277 (1159 vector sets)
Nimda-Normal	12,338	14,805 (2615 vector sets)
CodeRed I-Normal	9948	10,443 (1340 vector sets)

5.1. Decision tree analysis

In this section, Tables 6, 7, and 8 present four decision tree models, which are categorized as supervised machine learning algorithms [20]. CART is the classification and regression tree model, CIT is conditional inference tree, RF is random forest, and GBM is gradient boosting model. The results were obtained by using the datasets and after testing all the mentioned vector sets. We got improvements in ACC, FN, and PN after using the guide features, as shown in the tables for the three types of attacks, CodeRed-I, Nimda, and Slammer. Moreover, when comparing the results in the three tables we found that CIT provided a better rate of increase in accuracy (ACC) when compared with the results without GFG guide features, and the improvement rates were 10.96%, 13.39%, and 6.81% for CodeRed-I, Nimda, and Slammer, respectively. Besides, when we compared our work results with the GFG guide features to the results without GFG guide features, both CIT and RF gave better decrease rates with FN by 8.09% and 6.07% and by 3.4% and 3.86%, respectively, for CodeRed-I and Salmmer, respectively. With Nimda, both CIT and GBM gave better decrease rates with FN by 7.03% and 6.29%, respectively. It is clear that in Nimda attacks, the better rates occurred (CART and RF with FN) because the Nimda instances, which were used in training, are relatively close to normal instances. Also, CART presented the best improvement rates with FP especially for CodeRed-I (39.32%) and Slammer (29.81%) but gave the lowest improvement rates for FN in all cases of attacks. We concluded that CIT is clearly the best choice, especially for measuring ACC and FN, because it offers several advantages over other approaches. First, feature selection is unbiased (the traditional methods are biased towards features with many possible splits). Second, there is no need to 'prune' the resulting tree to avoid overfitting. Third, the algorithm returns values that show how confident one can be about every split.

Table 6. Decision trees with CodeRed-I attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
CART	0.188	0.8451	0.0544	0.6367	0.09	0.9365	0.0259	0.2435
CIT	0.154	0.8275	0.1061	0.4903	0.082	0.9371	0.0252	0.2434
RF	1.942	0.8915	0.0829	0.2485	1.258	0.9394	0.0222	0.2457
GBM	5.846	0.8431	0.0737	0.5452	5.9	0.9376	0.0243	0.2440

Table 7. Decision trees with Nimda attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
CART	0.248	0.6077	0.2425	0.4992	0.266	0.6914	0.2551	0.3836
CIT	0.106	0.5851	0.2399	0.5398	0.15	0.7190	0.1696	0.4371
RF	2.596	0.7156	0.0589	0.4422	2.906	0.7817	0.2574	0.1639
GBM	7.438	0.6310	0.2597	0.4456	7.574	0.7162	0.1968	0.4058

Table 8. Decision trees with Slammer attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
CART	0.132	0.8998	0.0512	0.3592	0.058	0.9675	0.0271	0.0611
CIT	0.11	0.8995	0.0606	0.3109	0.068	0.9676	0.0266	0.0629
RF	1.35	0.9195	0.0569	0.1997	0.94	0.9747	0.0183	0.0611
GBM	5.188	0.9049	0.0545	0.3109	5.214	0.9693	0.0247	0.0629

5.2. Neural network analysis

In this section, Tables 9, 10, and 11 present two neural network models, which are categorized as supervised machine learning algorithms [21]. ELM is extreme learning machine and SGD is stochastic gradient descent. We noticed improvements in ACC, FN, and PN rates after using the guide features, as shown in the tables for the three types of attacks. Also, it is obvious that SGD achieved better results compared to ELM. In addition, when we compared our results with the GFG guide features to the results without GFG guide features for the three types of attacks, we found that SGD provided better increase in ACC rates (between 9.76% and 32.36%), better decrease in FN rates (between 9.51% and 32.12%), and better decrease in FP rates (reaching 31.06%). As happened before with Nimda attacks, the better rates occurred (ELM and SGD with FP) because the instances used in training were relatively close to normal instances. We concluded that SGD is clearly the best choice not only as it gave the best increased rates in results but also because it offers fast convergence, efficiency, and ease of implementation and we observed that it does well with time-series data.

Table 9. Neural network models with CodeRed-I attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
ELM	2.25	0.8073	0.1043	0.5746	2.95	0.8255	0.0884	0.4864
SGD	1.52	0.7804	0.1260	0.5419	0.144	0.9338	0.0309	0.2313

Table 10. Neural network models with Nimda attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
ELM	2.49	0.6079	0.3834	0.3844	3.4	0.6209	0.3334	0.4149
SGD	1.518	0.5741	0.6119	0.2332	1.532	0.6717	0.4278	0.2654

Table 11. Neural network models with Slammer attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
ELM	1.974	0.8689	0.0825	0.3769	2.588	0.8788	0.0676	0.3305
SGD	1.528	0.6406	0.3536	0.2365	0.112	0.9642	0.0324	0.0538

5.3. SVM and BN analysis

In this section, we present the analysis results when using SVM and BN machine learning models. We discovered improvements in ACC, FN, and PN rates after using the guide features for the three types of attacks. Tables 12, 13, and 14 show these results. When we compared our results with the GFG guide features to the results without GFG guide features for the three types of attacks, we saw that SVM provided better increase in ACC rates (between 10.02% and 19.72%) and better decrease in FN rates (between 5.05% and 35.44%). BN gave ACC rates between 6.4% and 10.07% and a better decrease in FP rates (between 16.37% and 43.99%). Again, like with Nimda attacks, the better rates occurred (SVM with FP and BN with FN) because the instances used in training were relatively close to normal instances. Although SVM achieved high improvement rates, it has high computational complexity because of the quadratic optimization problem that needs to be solved and it consumes more time. We recommend that BN be the best choice regardless of the fact that it offers fewer improvements than SVM, especially when dealing with time-series data.

Table 12. SVM and BN with CodeRed-I attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
SVM	13.29	0.8374	0.0756	0.5796	2.814	0.9376	0.0251	0.2413
BN	0.012	0.8353	0.0574	0.6789	0.012	0.9360	0.0274	0.2390

5.4. Comparisons with feature selection algorithms

In this section, we compared the proposed ASMDF algorithm employing underlying cosine similarity with existing feature selection algorithms [22] by calculating the average processing time to select the dominant features from the vector set, as shown in Table 15. The results were obtained by using the datasets and after

Table 13. SVM and BN with Nimda attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
SVM	30.71	0.6068	0.4942	0.2517	27.77	0.7214	0.1398	0.4732
BN	0.006	0.6366	0.0267	0.8352	0.008	0.7006	0.0338	0.6715

Table 14. SVM and BN with Slammer attack.

	Without GFG guide features				With GFG guide features			
	T.sec	ACC	FN	FP	T.sec	ACC	FN	FP
SVM	6.256	0.7693	0.2228	0.2724	1.102	0.9665	0.0282	0.0611
BN	0.002	0.8863	0.0565	0.4160	0.006	0.9596	0.0387	0.0495

testing all the mentioned vector sets. Also, we used the GFG algorithm to generate the vector sets and replaced the proposed ASMDF algorithm with these feature selection algorithms one by one to measure the differences in processing time among them. These algorithms are CFS, symmetrical Uncertainty (SU), information gain (IG), gain ratio (GR), random forest (RF), and relief. We used the same simulation conditions for all these algorithms and calculated the average processing time for selecting the dominant features for the vector set received by these algorithms. As shown in Table 15, the proposed ASMDF algorithm is faster than SU by 3 times at least. Moreover, it is faster than the traditional RF by 8 times. CFS is faster than ASMDF, but feature selection using all other types including our proposed algorithm provided better dominant feature selections.

Table 15. The average processing time (s) for selecting the dominant features.

	CFS	ASMDF	SU	GR	IG	RF	Relief
CodeRed-I	0.0048	0.0094	0.0284	0.0286	0.0293	0.0784	0.6079
Nimda	0.0021	0.0049	0.0149	0.0157	0.0148	0.0399	0.2762
Slammer	0.0041	0.0093	0.0281	0.0278	0.0280	0.0737	0.5138

6. Conclusion

In this paper, we introduced a new idea for improving anomaly detection in time-series data by new guide features and feature selection. Previous studies usually focused on “inventing” more intricate models for different learning tasks. However, in some real-time systems, this complication makes hard for these models to work well. Our paper provides an alternative perspective by optimizing the feature set, which is good for improving the prediction and performance at the same time. To achieve this, we proposed two algorithms, GFG and ASMDF, in order to help detect and classify anomalies and to improve the performance of the anomaly detection process in time-series data. Moreover, we examined our proposed algorithms using several types of machine learning techniques and concluded that it has shown very promising feasibility for practical use and the GFG algorithm has the potential of detecting real outliers in noisy datasets. One of the most important additions of our research compared to much of the research in this field is the need to address and improve the features before being exploited. As a future work, we will focus on other imbalance types and try to make use of suitable techniques and therefore study the differences between anomalies and other imbalance types of traffic using our proposed algorithms.

References

[1] Al-Musawi B, Branch P, Armitage G. BGP anomaly detection techniques: a survey. IEEE Commun Surv Tut 2017; 19: 377-396.

[2] Ho TK. The random subspace method for constructing decision forests. IEEE T Pattern Anal 1998; 20: 832-844.

[3] Sidorov G, Gelbukh A, Gómez-Adorno H, Pinto D. Soft similarity and soft cosine measure: similarity of features in vector space model. Computación y Sistemas 2014; 18: 491-504.

[4] Chandola V, Banerjee A, Kumar V. Anomaly detection: a survey. ACM Comput Surv 2009; 41: 1-58.

[5] Hajji H. Statistical analysis of network traffic for adaptive faults detection. IEEE T Neural Networ 2005; 16: 1053-1063.

[6] Thottan M, Liu G. Anomaly detection approaches for communication networks. In: Cormode G, Thottan M, editors. Algorithms for Next Generation Networks. London, UK: Springer, 2010. pp. 239-261.

[7] Tan P, Steinbach M, Kumar V. Introduction to Data Mining. 1st ed. Boston, MA, USA: Addison-Wesley, 2005.

[8] Augusteijn M, Folkert B. Neural network classification and novelty detection. Int J Remote Sens 2002; 23: 2891-2902.

[9] Diaz I, Hollmen J. Residual generation and visualization for understanding novel process conditions. In: IEEE IJCNN'02 Neural Networks Conference; 12–17 May 2002; Honolulu, HI, USA. New York, NY, USA: IEEE. pp. 2070-2075.

[10] Sharma O, Girolami M, Sventek J. Detecting worm variants using machine learning. In: Proceedings of CoNEXT Conference; 10–13 December 2007; New York, NY, USA. New York, NY, USA: ACM. pp. 1-12.

[11] Moore A, Zuev D. Internet traffic classification using Bayesian analysis techniques. In: Proceedings Conference on Measurement and Modeling of Computer Systems; 6–10 June 2005; Alberta, Canada. New York, NY, USA: ACM. pp. 50-60.

[12] El-Arini K, Killourhy K. Bayesian detection of router configuration anomalies. In: Proceedings of Workshop on Mining Network Data; 26 August 2005; Philadelphia, PA, USA. New York, NY, USA: ACM. pp.221-222.

[13] Wubbeling M, Elsner T, Meier M. Inter-AS routing anomalies: improved detection and classification. In: IEEE 6th International Conference On Cyber Conflict; 3–6 June 2014; Tallinn, Estonia. New York, NY, USA: IEEE. pp. 223-238.

[14] Deshpande S, Thottan M, Sikdar B. Early detection of BGP instabilities resulting from Internet worm attacks. In: IEEE GLOBECOM'04 Global Telecommunications Conference; 29 November–3 December 2004; Dallas, TX, USA. New York, NY, USA: IEEE. pp. 2266-2270.

[15] Bilski T. Disaster's impact on Internet performance – case study. In: Bilski T, editor. Communications in Computer and Information Science. Heidelberg, Germany: Springer, 2009. pp. 210-217.

[16] Makridakis G, Wheelwright C, Hyndman J. Forecasting: Methods and Applications. 3rd ed. New York, NY, USA: Wiley, 1997.

[17] Cohen J, Cohen P, West G, Aiken S. Applied Multiple Regression Correlation Analysis for the Behavioral Sciences. 3rd ed. Mahwah, NJ, USA: Lawrence Erlbaum Associates, 2003.

[18] Draper R, Smith H. Applied Regression Analysis. 3rd ed. New York, NY, USA: Wiley, 1998.

[19] Ladha L, Deepa T. Feature selection methods and algorithms. International Journal on Computer Science and Engineering 2011; 3: 1787-1797.

[20] Strobl C, Malley J, Tutz G. An introduction to recursive partitioning: rationale, application and characteristics of classification and regression trees, bagging and random forests. Psychol Methods 2009; 14: 323-348.

[21] Weibo L, Zidong W, Xiaohui L, Nianyin Z, Yurong L, Alsaadi F. A survey of deep neural network architectures and their applications. Neurocomputing 2017; 234: 11-26.

[22] Jundong L, Kewei C, Suhang W, Fred M, Robert P, Jiliang T, Huan L. Feature selection: a data perspective. ACM Comput Surv 2010; 9: 1-45.