Suitability of Graph Representation for BGP Anomaly Detection

Kevin Hoarau, Pierre Ugo Tournoux, Tahiry Razafindralambo

Suitability of Graph Representation for BGP Anomaly Detection

Kevin Hoarau
Université de La Réunion, LIM, France
kevin.hoarau@univ-reunion.fr

Pierre Ugo Tournoux
Université de La Réunion, LIM, France
pierre.tournoux@univ-reunion.fr

Tahiry Razafindralambo
Université de La Réunion, LIM, France
tahiry.razafindralambo@univ-reunion.fr

***Abstract*—The Border Gateway Protocol (BGP) is in charge of the route exchange at the Internet scale. Anomalies in BGP can have several causes (mis-configuration, outage and attacks). These anomalies are classified into large or small scale anomalies. Machine learning models are used to analyze and detect anomalies from the complex data extracted from BGP behavior. Two types of data representation can be used inside the machine learning models: a graph representation of the network (graph features) or a statistical computation on the data (statistical features). In this paper, we evaluate and compare the accuracy of machine learning models using graph features and statistical features on both large and small scale BGP anomalies. We show that statistical features have better accuracy for large scale anomalies, and graph features increase the detection accuracy by 15% for small scale anomalies and are well suited for BGP small scale anomaly detection.**

***Index Terms*—BGP Anomaly, Machine Learning, Graph.**

I. Introduction

The Border Gateway Protocol (BGP) is the routing protocol standard on the Internet. A failure of the protocol could impact any service relying on the Internet. Such failures, namely BGP anomalies, happen for several reasons ranging from hardware failures to malicious attacks [1]. BGP anomalies and their detection are studied using BGP data collection projects such as [15], [16]. These anomalies can be classified as large scale and small scale based on the number of route affected and their consequences on the Internet. At the Internet scale, BGP data traces are complex and require advanced techniques such as machine learning to be analyzed in order to detect anomalies. BGP data traces are processed and transformed into statistical features (*e.g.* counting the number of announcements, prefixes) or into graph features, *i.e.* metrics derived from the BGP graph. Machine learning models for BGP anomaly detection can be fed using graph features or statistical features extracted from BGP data traces.

Statistical features have long been shown to be effective to detect large scale BGP anomalies. The effectiveness of graph features for that purpose has only recently been shown [18]. On one hand, large scale anomalies seem easier to detect due to their impacts and consequences on the Internet. On the other hand, small scale anomalies seem more difficult to detect since they can be localized and their impact may take time to propagate on the Internet. However, it is not clear which feature (graph or statistical) is better than the other in detecting large scale and small scale anomalies when plugged into machine learning models.

Results from the literature show two different types of study: 1) anomaly classification of known events and 2) anomaly detection. We focus on the former one. The authors of [2] use machine learning models to detect large scale events with statistical features and show the efficiency of the method. In [18], the authors study large scale events detection using graph features plugged into machine learning model and show that graph features have good performances. In this paper, we fill the gap in the literature by providing a performance comparison of machine learning models using graph features and statistical features for small scale and large scale BGP anomaly detection.

To detect small scale and large scale BGP anomalies we use machine learning models such as support-vectors machines, multi-layer perceptron, naive Bayes, decision tree and k-nearest neighbors. These anomalies are detected by either using graph features or statistical features inside the machine learning model.

In this paper we show that: i) for large scale anomaly events, statistical features have better performance than graph features but graph features provide good accuracy rate when used in machine learning models; ii) the detection accuracy of small scale events is improved by 15% when using graph features instead of statistical features. These results are promising since with a fine tuned machine learning model and a larger training set, the detection accuracy of small scale BGP anomaly detection could be further increased.

The reminder of this paper is organized as follows. In section II we provide a background on BGP, a description of BGP anomalies and a state of the art summarizing the use of machine learning in BGP anomaly detection. In section III we describe the data set used in the paper. Section IV is devoted to the data analysis while section V is focused on the evaluation of machine learning models' accuracy for anomaly detection. We conclude and discuss the results of the paper in section VI.

II. Background and related work

A. BGP in a nutshell

The Internet consists of Autonomous Systems (ASes) interconnected by Border Gateway Protocol (BGP). Most of the

This project has received funding from the Région Réunion and the European Union - European Regional Development Fund (ERDF) as part of the INTERREG V - 2014-2020 program.

ASes are Internet Service Providers (identified by an ASN) that own IP prefixes [10]. ISPs operate BGP routers that maintain TCP connections with a set of BGP neighbors to exchange routing information with other ASes. Traffic is sent through routes learned by BGP. BGP incrementally updates its routes. When using BGP, a route to an IP prefix is identified by the set of ASes (namely the AS-PATH) that participate in the traffic forwarding which avoids routing loops [20].

B. BGP Anomalies

In this work, we distinguish three types of BGP anomalies.

1) Large scale anomalies: these types of anomalies have a major impact both on BGP and Internet's data plan. While they are easier to address from an anomaly detection perspective, their impact is such that several works focus on their detection. These anomalies [4] are mostly due to configuration errors [14], worm spread [19], power outage [7] or hardware failure [5].

2) Small scale origin hijacking: this type of anomalies happens when an AS announce a prefix it does not own [3]. As BGP routers favor shorter paths, other ASes may choose the illegitimate route if the path to the hijacker is shorter than the path to the legitimate origin. These events often result in the apparition of a Multiple Origin AS [21] (MOAS) conflict which makes them more noticeable. However, MOAS conflict can also happen due to legitimate practices as multi-homing and prefix reallocation. Attackers can also announce a sub-prefix to avoid the apparition of a MOAS conflict. Therefore, the distinguishing between legitimate and malicious MOAS requires more advanced approaches.

3) Small scale path hijacking: this type of anomalies occurs when an AS forges a path for a prefix [3]. A forged path might be smaller than the legitimate path and results in the traffic destinated to the origin being directed toward the hijacker. These anomalies are even more subtle to detect than origin hijacking as the origin AS stays untouched. If the hijacker AS does not have a valid route for the prefix the consequence on the data plane is a traffic black-holing. However, if the hijacker is able to redirect the traffic to the origin AS, then the hijacker could observe and modify the traffic without anyone noticing.

C. BGP anomaly detection using Machine Learning

The collection of BGP routing information is the cornerstone for any analysis of the BGP protocol. RouteViews [16] and RIPE RIS [15] projects have been collecting and archiving BGP data from different collectors distributed across the world since 2000. Each of these collectors receives and saves BGP updates from all its neighboring routers and updates its Routing Information Base (RIB) accordingly.

1) Statistical features: The machine learning models for BGP anomaly detection does not consume raw BGP data from RouteViews and RIPE RIS. They are transformed in statistical features which can be classified as i) volume features such as the number of announcements and withdrawals which aim to capture changes in the stability of BGP; ii) AS-PATH features such as average AS-PATH length and the maximum edit distance aim to capture topological changes.

Various ML algorithms have been used to process these features *e.g.* SVM [8], [6], Naive Bayes classifier [8], [6], decision trees [6], [13] and more recently deep learning [8], [2], [13], [4]. These works achieved good performance on the detection of large-scale anomalies such as worms spread, massive route leaks, large-scale power outage, and submarine cable cut.

2) Graph features: More recently, some authors chose to leverage the underlying graph structure of BGP instead of the statistical features [18], [11]. These dynamic graphs reflect the evolution of the BGP topology where ASes are the graph's nodes and adjacent AS in AS-PATH are the graph's edges. In [18], metrics from the graph theory such as centrality metrics are used as features for an ML model. Plus, the results provide some evidence that graph metrics such as the clustering coefficient may be used to detect small-scale events. In [11], Goyal et al. introduce a graph embedding algorithm designed to generate stable embeddings of dynamic graphs. The embedding was applied to BGP data and the result shows that changes in the embedding seem to be correlated with anomaly events.

However, as far as we know, there is no fair comparison of statistical and graph features in the literature. Our goal is to fill this gap and to identify if machine learning based BGP anomaly detector can benefit from graph feature instead of the widely adopted statistical features.

III. Dataset

Our dataset includes 4 large scale anomalies as well as 17 occurrences of origin hijacking and 14 occurrences of path hijacking. Each of these events includes 2 hours of BGP data which are sampled every two minutes to generate a total of 60 BGP snapshot per event. From every snapshot, we extract 32 statistical features and 31 graph features. This results in a dataset suitable for the comparison of statistical and graph features, containing a total of 2100 time series. The remaining of this section details the events included in our dataset, the data collection process and the extracted features.

A. BGP anomaly events

We chose to rely on well documented BGP events including 4 large scale events, 17 small scale origin hijacking events and 14 small scale path hijacking events that were reported in a blog post [9]. The origin and path hijacking events have been used in [3] for the purpose of anomaly classification while the large scale events have been used in [18] for the detection of BGP anomaly using graph features. These events have been thoroughly reviewed and can safely be considered as ground truth.

B. Data collection

For both data collection and features extraction we use BML [12]. For all the events, we collect data one hour before and one hour after the estimated start of the event. Therefore,

for each event we use 2 hours of BGP data. The data are collected from the `rrc04` and `rrc05` collectors located in Geneva and Vienna. These collectors were chosen for their intensive use in previous research [2], [4], [18]. BGP being an incremental protocol, we need to collect data during a priming period before the 2 hours time window. Using Ripe RIS collectors, RIB dumps are available every 8 hours. So we used a priming period of 10 hours to ensure that at least one RIB dump is collected which allows us to have a complete view of the routes available on a collector. The update messages received between the RIB dump and the observation window are used to update the routes. Thankfully, all this work is automatically carried out by BML [12].

C. Features extraction

For all the events in our dataset we used BML to extract 32 BGP statistical features and 31 graph features every 2 minutes which gives us 60 samples of 63 features per event. The BGP statistical features which are computed from the update messages collected within the 2 minutes interval are usually divided in two categories [1], [2], [4]:

- Volume features: these features are computed on the volume of the update messages collected. BML allows to compute 15 volume features, as the number of route announcements and withdrawals, the number of origin changes, the average number of announcements per AS and inter-arrival time of BGP updates.
- AS-path features are computed using the AS-path field of a route announcement. Examples of these features are the number of announcements to a shorter or to a longer AS-path and the AS-path edit-distance. In total, 17 AS-path features are computed.

For the graph feature extraction, BML first extracts the topology of the BGP network using the as-path and the routes available at the beginning of the time windows. Secondly, the topology is updated every 2 minutes using the collected update messages. Finally, based on graph theory metrics, 31 graph features are extracted. These metrics are divided into two categories:

- Node level metrics: these metrics are computed for all the nodes of the graph and we use the average value as a feature. This metrics includes centrality metrics which measure the importance of a node in a graph. Metrics such as the clustering coefficient, eccentricity and the degree which measure the connectivity of a node are also extracted. In total, 16 node level metrics are computed.
- Graph level metrics: are globally defined and give a single value for an entire graph. These metrics have already been used to evaluate the structural robustness of a network [17]. We computed 15 graph level metrics among which the algebraic connectivity, the weighted spectrum and the percolation limit.

The computation of these graph metrics can be both processor and memory consuming, especially on large graphs (BGP network with approximately 54K nodes for data collected in 2016). For the purpose of computing these graph metrics in a reasonable amount of time in our environment setup[1], we choose to reduce the dimension of the BGP network by extracting the k-core of the graph. In graph theory the k-core of a graph is a maximal subgraph that contains nodes of degree k or more. Our motivation was to extract the most connected portion of the graph as we assume that this portion should support most of the traffic and therefore being the most critical. However, as we are aware that this dimension reduction of the graph induces a significant loss of information, we also extract a subset of the graph feature from the original BGP network. This subset is composed of 17 graph features that we were able to compute in a reasonable amount of time in our setup. It allows us to evaluate the impact of the k core extraction on these metrics.

A recurrent issue when working with multiple features is having features that do not change inside the same range. This can result in bias in the subsequent data analysis by giving more weights to some features. To prevent this, we normalized all the features using z-score normalization which transforms the features into a zero mean and unit variance distribution.

D. Data labeling

When visualizing data or training a machine learning model it is useful to have label associated with the data. In our case, for an event in our dataset consisting of 60 samples, we have a binary vector of size 60 where a sample is labeled ”1” if it was extracted during the anomaly time. For each event, we collected one hour of data before the event and one hour of data after the start time of the event. We use a simple approach where we labeled the second half of the event as anomalous (after the start time of the event). This results in a label vector of size 60 where the last 30 values are ”1”.

It is important to notice that, for the origin and path hijacking events, we do not have an exact start time for the events. Thus, when visualizing the data we may expect a time shift between the label and the apparition of the anomaly. Furthermore, this inaccuracy in the labeling process could affect the training of supervised machine learning models.

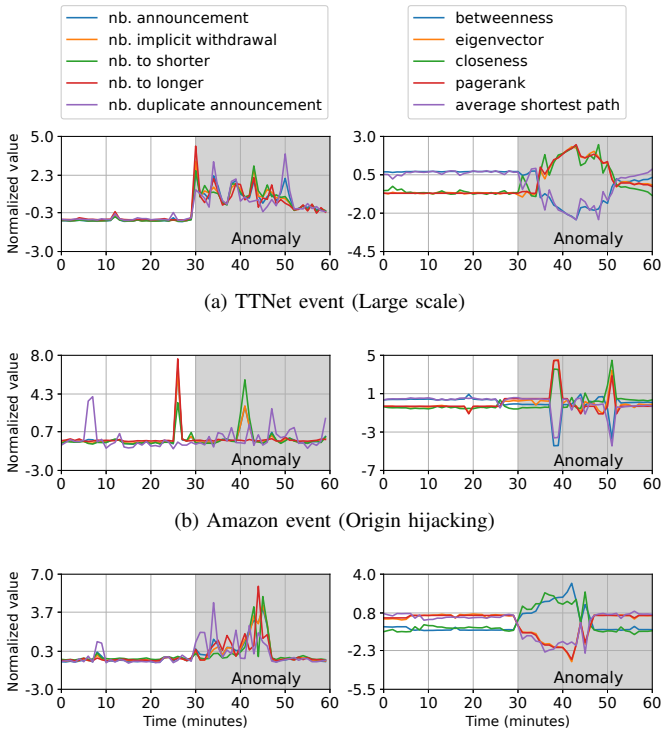
IV. Data analysis

A. Raw features visualisation

We first visualize the raw features to have some insights about the impact of an anomaly on the statistical and graph features. Due to space constraints, we pick one event per type of anomaly and we keep 5 statistical and graph features. Both the events and the features were manually selected for the visibility of the impact. Therefore, they must be seen as the best case scenario and not necessarily representative of the other samples in the dataset. Figure 1 shows the selected statistical and graph features for 3 events. On the large scale event [18], the anomaly is clearly visible with both statistical and graph features. However, for the origin and path hijacking

[1] During our test case, we used a computer with the following specification: 3.7GHz/8 cores processor with 64Gb of RAM running Ubuntu 18.04

events [9] the anomalies are more visible with graph features whereas statistical features are more noisy. On the origin hijacking event, we can see a delay between the start time of the anomaly and the spikes in the features. This is probably due to the inaccurate labeling for this type of event.



(a) TTNet event (Large scale)

(b) Amazon event (Origin hijacking)

(c) France event (Path hijacking)

Fig. 1. Statistical features (Left) and Graph features (Right)

B. *Synthetic visualisation - PCA*

We use a principal component analysis (PCA) to reduce the dimensionality of the data since visualizing and analyzing the raw features is difficult when their number is high. For each event in the dataset, the PCA is applied on both statistical and graph features. By keeping only the first and second principal component, we obtain 60 samples in 2 dimensions. Figure 2 shows PCA 2d projection of the statistical features (left subfigure) and graph features (right subfigure) during an origin hijacking event. We can see that for the graph features, the samples extracted before the anomaly (with a "0" label) are grouped together and well separated from the data extracted during the anomaly. The figure also suggests that the statistical features may not easily distinguish BGP's behaviour before and during the event.

To measure the separation between anomalous and non-anomalous data we used the silhouette coefficient which is commonly used to evaluate the partition of a clustering algorithm. For each point in the data, the silhouette coefficient evaluates how near the point is to the points from the same cluster and how far it is from the points from other clusters.

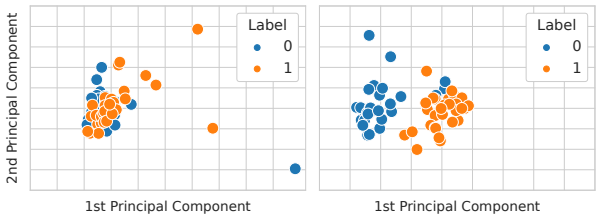


Fig. 2. 2d projection of statistical (Left) and graph (Right) features during amazon event (Origin hijacking)

The silhouette coefficient of a point i is then defined as:

$$S(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}} \quad (1)$$

where $a(i)$ is the mean distance between i and all points from the same cluster and $b(i)$ is the smallest mean distance of i and the points in all the other clusters. The silhouette coefficient for the partition is then computed as the mean value over all the points. The silhouette coefficient ranges from -1 to 1 where 1 is the best value, 0 is obtained on overlapping clusters and -1 when data has been assigned to the wrong clusters. As the clustering coefficient can be computed in the original feature space, we could have skipped the dimension reduction step using PCA. However, in our experiments this lead to lower values of the silhouette coefficient. This is due to some noisy features whose effects are mitigated by the dimension reduction. When we compute the silhouette coefficient for the data showed in figure 2 we obtain a value of 0.52 for the graph features which looks reasonable as the data is quite well separated and 0.01 for the statistical feature as the data overlaps.

The silhouette coefficient gives us a single value to evaluate how well the data from an event is separated. Therefore, we can use it as a metric to compare the separability of statistical and graph features. We computed the silhouette coefficient for all the events of the dataset for both types of features. Figure 3 shows the distribution of the silhouette coefficient when the values are grouped by type of events. We can see from figure 3 that the data is slightly better separated using statistical features for the large scale events even if the value for graph feature is still above 0.4. However, for both origin and path hijacking, the silhouette coefficients for statistical features are close to 0 meaning that the clusters are overlapping. Graph features give better results with a mean value around 0.2. These results could probably be improved with a more accurate labeling of the hijacking events. From these results, we can expect to achieve slightly better performance using statistical features instead of graph features for the detection of large scale events. However, for both origin and path hijacking, detection should be nearly random using statistical features while graph features may achieve better results.

V. ML BASED ANOMALY DETECTION

In this section, we compare the benefits of statistical and graph features for the detection of BGP anomalies. We build

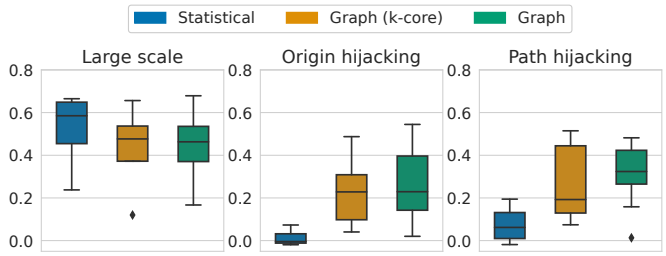


Fig. 3. Silhouette coefficient

a machine learning pipeline to classify a feature sample as anomalous or non-anomalous.

A. Experiment setup

Our machine learning pipeline is implemented using the *scikit-learn* library. The implementations are available online[2].

1) ML algorithms: Different machine learning algorithms can achieve various results on the same classification task, we evaluate the performance of our pipeline on 5 machine learning algorithms. We used well-known machine learning algorithms that have already been used for the detection of BGP anomalies based on both statistical and graph features [2], [6], [18]: a support-vector machine (SVM) with a radial basis function kernel, a multi-layer perceptron (MLP) with 3 hidden layers, a naive Bayes (NB) classifier, a decision tree (DT) and a k-nearest neighbors (KNN) classifier.

2) Dimensionality reduction: A good practice when training a ML model is to reduce the dimension of the feature space. This allows us to filter out the redundant and irrelevant information. Moreover, this prevents over-fitting and avoids the curse of dimensionality which could be an issue due to the limited size of our dataset. We used a principal component analysis (PCA) to project the features in a 2 dimension space as described in section IV.

3) Cross validation: To evaluate the performance of our model we used a cross validation approach with a Leave One Group Out (LOGO) scheme. For a dataset composed of n events, the model is trained using the data from $n-1$ events and the remaining sample is used as an independent test set. In a round-robin fashion, each event is used as a test and the performance can be averaged across the n iterations. This approach allows us to use a large portion of the data in the training set while producing statistically robust results. In our experiments we used the *accuracy* to evaluate the performance of a model:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

where TP, TN, FP and FN come from the confusion matrix.

4) Hyper-parameters tuning: Having multiple ML algorithms implies to deal with more hyper-parameters. These hyper-parameters are, for instance, the learning rate of the MLP, the maximum depth of the DT or the number of neighboors for the KNN. To find the best combination of parameters for each ML algorithm we used the *GridSearchCV* function of *scikit-learn*. This function allows us to define a grid of hyper-parameters values for each ML algorithm and searches exhaustively the best combination from this grid. For each possible combination, a cross-validation is done on the training set to avoid over-fitting. For this step, we used a 3-fold group cross validation that divides the events in 3 subsets. For an iteration, the data from 2 subsets are used for training and the remaining subset is used for testing.

[2]https://github.com/KevinHoarau/BGP_stat_vs_graph

B. Performance comparison

To evaluate the performance of the models we applied our ML pipeline separately for each type of anomaly. Figure 4 shows the performance of the 5 models. As expected from previous sections, the statistical features outperforms the graph features on the large scale events (95% accuracy using the NB model) but performs near to a random classifier on origin and path hijacking events. However, graph features on large scale event achieved 90% accuracy on large scale event using the k-core and 85% otherwise. On origin and path hijacking events the graph features (without k-core) is the best option as 60% accuracy is obtained using the SVM model. Based on these results, we used the graph features without k-core as it gives stable results across the different types of events compared to the k-core option.

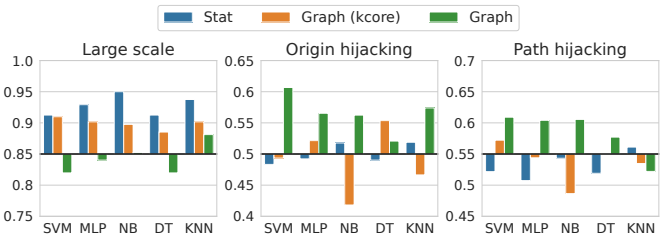


Fig. 4. Accuracy (see eq. 2) of ML models

C. Anomaly detection

Figure 5 shows the decision boundary of the SVM model on an origin and a path hijacking event using graph features. We only focus on the results from the SVM model since it provides the best performances. Thanks to the PCA for the dimensionality reduction the data can be plotted in 2 dimensions. We can see from figure 5 that some blue points are in the orange section meaning that they are false positives. This can be the consequence of the inaccurate labelling process of this data. False positives can also occur due to a misclassification of the model. To mitigate this issue, we can use multiple data points and detect an anomaly only if a significant portion of these points are classified as anomalous. To evaluate the performance of such anomaly detection scheme, we divide each event in two subsets: the negative and positive set. The negative (resp. positive) subset is composed of the 30 points labelled non-anomalous (resp. anomalous). Then, for each

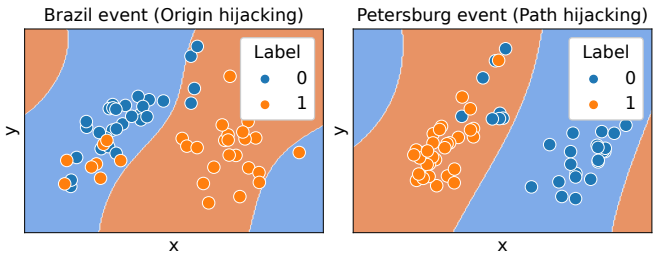


Fig. 5. Decision boundary of the SVM model on two hijacking events

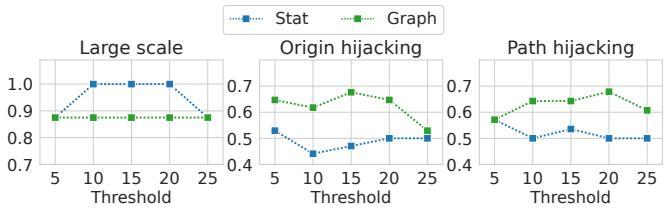


Fig. 6. Anomaly detection accuracy (see eq. 2) as a function of the threshold *i.e.* the required number of samples to be classifed as anomaly by the SVM.

subset, all the data points are classified by the ML model and an anomaly is detected only if the number of points classified as anomalous is above a threshold.

Figure 6 shows the performance of the SVM model for anomaly detection depending on the threshold. We can see from this figure that the best performances are achieved from 15 to 20 events as threshold. On one hand, for the large scale events using statistical features we obtain 100% accuracy against 88% for the graph features. On the other hand, for the origin and path hijacking events, the graph features give 68% accuracy while for the statistical features the performances are equivalent to a random classifier (around 50%).

VI. Conclusion

This paper evaluates and compares the accuracy of machine learning models for anomaly detection in BGP. Especially, we use different data representations to describe the behaviour of BGP : a graph representation (graph features) and a statistical representation (statistical features). We plug the different data representations inside a machine learning model and evaluate the anomaly detection accuracy of the model. This accuracy is evaluated on large scale and small scale anomaly events.

We show that for large scale anomalies, statistical features have better detection accuracy than graph features. However, machine learning models used with graph features provide a decent accuracy. We also show that while none of the types of features provides satisfying performances for small scale events detection, the graph features increase the detection accuracy by 15% compared to statistical features. These preliminary results on small scale BGP anomalies are promising as with a fine tuned machine learning model and a larger training dataset, the detection accuracy may be further increased resulting in satisfying performances.

References

[1] B. Al-Musawi, P. Branch, and G. Armitage, "BGP anomaly detection techniques: A survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 377–396, FebJan 2017. [Online]. Available: http://dx.doi.org/10.1109/comst.2016.2622240

[2] M. Cheng, Q. Li, J. Lv, W. Liu, and J. Wang, "Multi-scale lstm model for bgp anomaly classification," *IEEE Transactions on Services Computing*, 2018.

[3] S. Cho, R. Fontugne, K. Cho, A. Dainotti, and P. Gill, "Bgp hijacking classification," in *2019 Network Traffic Measurement and Analysis Conference (TMA)*, 2019, pp. 25–32.

[4] M. Cosovic, S. Obradovic, and E. Junuz, "Deep learning for detection of bgp anomalies," in *International Work-Conference on Time Series Analysis*. Springer, 2017, pp. 95–113.

[5] J. H. Cowie, A. T. Ogielski, B. Premore, E. A. Smith, and T. Underwood, "Impact of the 2003 blackouts on internet communications," *Preliminary Report, Renesys Corporation (updated March 1, 2004)*, 2003.

[6] I. O. de Urbina Cazenave, E. Köşlük, and M. C. Ganiz, "An anomaly detection framework for bgp," in *2011 International Symposium on Innovations in Intelligent Systems and Applications*, June 2011, pp. 107–111.

[7] S. Deshpande, T. Ho, M. Thottan, and B. Sikdar, "An online mechanism for bgp instability detection and analysis," *IEEE Transactions on Computers*, vol. 58, no. 11, pp. 1470–1484, nov 2009.

[8] Q. Ding, Z. Li, P. Batta, and L. Trajković, "Detecting bgp anomalies using machine learning techniques," in *2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2016, pp. 003 352–003 355.

[9] Dyn, "Oracle Internet Intelligence Blog." [Online]. Available: https://blogs.oracle.com/internetintelligence/

[10] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Transactions on Networking*, vol. 9, no. 6, pp. 733–745, Dec 2001.

[11] P. Goyal, N. Kamra, X. He, and Y. Liu, "Dyngem: Deep embedding method for dynamic graphs," *arXiv preprint arXiv:1805.11273*, 2018.

[12] K. Hoarau, P.-U. Tournoux, and T. Razafindralambo, "BML: an efficient and versatile tool for BGP dataset collection," in *WS22 IEEE ICC 2021 the 3rd International Workshop on Data Driven Intelligence for Networks and Systems (WS22 ICC'21 Workshop - DDINS)*, Montreal, Canada, Jun. 2021.

[13] Y. Li, H. J. Xing, Q. Hua, X. Z. Wang, P. Batta, S. Haeri, and L. Trajković, "Classification of bgp anomalies using decision trees and fuzzy rough sets," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2014, pp. 1312–1317.

[14] R. Mahajan, D. Wetherall, and T. Anderson, "Understanding bgp misconfiguration," *ACM SIGCOMM Computer Communication Review*, vol. 32, no. 4, pp. 3–16, 2002.

[15] RIPE, "Routing information service (RIS)." [Online]. Available: https://www.ripe.net/analyse/internet-measurements/routing-information-service-ris/routing-information-service-ris

[16] RouteViews, "Routeviews - university of oregon route views project." [Online]. Available: http://www.routeviews.org/routeviews/

[17] D. F. Rueda, E. Calle, and J. L. Marzo, "Robustness comparison of 15 real telecommunication networks: Structural and centrality measurements," *Journal of Network and Systems Management*, vol. 25, no. 2, pp. 269–289, 2017.

[18] O. R. Sanchez, S. Ferlin, C. Pelsser, and R. Bush, "Comparing machine learning algorithms for bgp anomaly detection using graph features," in *Proceedings of the 3rd ACM CoNEXT Workshop on Big DAta, Machine Learning and Artificial Intelligence for Data Communication Networks*, 2019, pp. 35–41.

[19] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, A. Mankin, S. F. Wu, and L. Zhang, "Observation and analysis of bgp behavior under stress," in *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement*, 2002, pp. 183–195.

[20] S. H. Y. Rekhter, T. Li, "A border gateway protocol 4 (bgp-4)," Network Working Group, IETF, RFC 4271, 2006. [Online]. Available: https://tools.ietf.org/html/rfc4271

[21] X. Zhao, D. Pei, L. Wang, D. Massey, A. Mankin, S. F. Wu, and L. Zhang, "An analysis of bgp multiple origin as (moas) conflicts," in *Proceedings of the 1st ACM SIGCOMM Workshop on Internet Measurement*, ser. IMW '01. New York, NY, USA: ACM, 2001, pp. 31–35. [Online]. Available: http://doi.acm.org/10.1145/505202.505207