ISP Self-Operated BGP Anomaly Detection Based on Weakly Supervised Learning

Yutao Dong[a,b], Qing Li[b*], Richard O. Sinnott[c], Yong Jiang[a,b] ,Shutao Xia[a,b]
[a] Tsinghua Shenzhen International Graduate School, Tsinghua University
[b] Peng Cheng Laboratory (PCL), [c] University of Melbourne
dyt20@mails.tsinghua.edu.cn, liq@pcl.ac.cn, rsinnott@unimelb.edu.au, {jiangy, xiast}@sz.tsinghua.edu.cn

2021 IEEE 29th International Conference on Network Protocols (ICNP) | 978-1-6654-4131-5/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICNP52444.2021.9651957

***Abstract*—The Border Gateway Protocol (BGP) is arguably the most important and irreplaceable protocol in the network. However, the lack of routing authentication and validation makes it vulnerable to attacks, including routing leaks, route hijacking, prefix hijacking, etc. Therefore, in this paper we propose a generalized framework for ISP self-operated BGP anomaly detection based on weakly supervised learning. To tackle the problem of insufficient data in BGP anomaly detection, we propose an approach to learn from the other anomaly detection systems through knowledge distillation. To reduce the impact of inaccurate supervision, we design a self-attention-based Long Short-Term Memory (LSTM) model to self-adaptively mine the differences between BGP anomaly categories, including both feature and time dimensions. Finally, we implement a system and demonstrate the performance through a set of comprehensive experiments. Compared with the state-of-the-art schemes, our scheme has better generalization on various anomaly types.**

***Index Terms*—BGP, Anomaly Detection, Self-operated, Weakly Supervised Learning**

I. Introduction

The Border Gateway Protocol (BGP) lacks routing authentication and validation, which makes the Internet more susceptible to malicious users and misconfigurations, such as prefix hijacking [1], routing leaks [2]–[4], breakouts [5]–[7] and route hijacking [8]. These attacks may cause significant economic loss, potential leak of user privacy and the inevitable interruption of large-scale networks, impacting thousands of users and organisations. For example, on August 31, 2020, CenturyLink's mis-configuration of BGP caused a 3.5% drop in global Internet traffic, which is one of the largest Internet outages in history [9]. In fact, an ever-increasing number of BGP anomalies and vulnerabilities have been discovered in the Internet [10]. Therefore, it is significantly important to provide an efficient scheme to identify these abnormal events.

The existing schemes on BGP security can be divided into security protocol protection and anomaly detection. The security protocol protection aims to prevent the anomalies by various routing authentication technologies (*e.g.*, RPKI [11] and S-BGP [12]). Nevertheless, these methods can not work until most Autonomous Systems (ASes) have deployed them. Unfortunately, due to issues such as upgrade costs, it is too difficult to change the Internet, especially when a single AS deployment cannot take effect [13]. Thus, Internet Service Providers (ISPs) prefer to use anomaly detection to defend against BGP anomalies. Meanwhile, the anomaly detection schemes can be further divided into two sub-categories: rule-based approaches (*e.g.* PHAS [14], [15], [16], ARTEMIS [17]) and machine learning based approaches (*e.g.*, Random Forest [18]–[20], SVM [21], MLP [22] and LSTM [23]–[25]). Generally, the rule-based approaches can only detect prefix hijacking and one-hop route hijacking. These approaches cannot detect new anomalies nor more sophisticated hijacking (*e.g.*, more than one hop hijacking, route leaks) due to the inflexibility of rules. In contrast, the machine learning-based approaches will get much better generalization performance on sophisticated hijacking. However, the existing learning-based approaches can only analyze and verify large-scale abnormal events, such as worm attacks and breakout incidents. These approaches which lack sufficient data for training cannot work on other smaller effect events, like route hijacking attacks.

In addition, most ISPs rely on third parties [5], [8], [26] to check whether their network is under attack [13], because no approach can handle all the anomalies. However, relying on third parties causes some issues. Firstly, once the network configuration is changed locally, they need to update the third-party anomaly detection systems in time. Secondly, an anomaly detection system is designed for some specific anomalies and cannot work effectively on the other anomalies, so none of the systems can detect all the anomalies. Therefore, the ISPs have to reach agreements with multiple third-party anomaly detection systems to detect all the anomalies. As such, the ISPs cannot react quickly enough to detect anomalies and avoid large-scale BGP oscillation.

As a result, we aim to design an anomaly detection system for an AS to identify all BGP anomalies without the cooperation of other ASes. The goal is to quickly detect the BGP anomalies then alleviate the negative impact. To achieve this objective, there are two main challenges to be addressed:

1) Since BGP abnormal events are difficult to collect and to verify due to the complicated commercial concerns, it is inevitable to learn by inaccurate supervision. Therefore, the first challenge is to build a sizable comprehensive weak anomaly dataset.
2) The given labels are not always verified ground truth [27] and the classes are imbalanced. Hence, the second challenge is to train efficient anomaly detection models from potentially inaccurate and imbalanced data.

In order to solve these challenges, we propose a general framework for BGP anomaly detection based on weakly supervised learning, which adapts to all BGP anomalies without manual intervention. Specifically, we put forward a knowledge distillation of anomaly detection systems to construct a sufficient dataset and use the de-noise model as a detection model. Thus, our work can be divided into two parts.

- **Knowledge distillation of anomaly detection systems:** We use the existing anomaly detection systems as a teacher system to construct sufficient anomaly events composed of thousands of actual events. To verify the usability of inaccurate data, we perform data analysis on massive BGP anomaly events, based on which we are able to confirm that different anomalies can be distinguished by characteristics of the data.
- **De-noise model:** To reduce the impact of inaccuracies, we use the LSTM model based on the self-attention mechanism as a student model. The self-attention method is used to mine the importance of time steps, while the LSTM mechanism is to find the importance of features. By using both methods we can minimize the impact of data noise. In addition, we design a dynamic stride sampling method to deal with the imbalance problem.

We implement a prototype of the distillation method using multiple anomaly detection systems such as [5] and [8]. The dataset and prototype have been published on GitHub[1]. The experiment results show that our scheme can identify all types of anomalies and achieve similar performance to the original specific anomaly detection system. We also use a public well-known anomaly dataset such as [28] to test our trained model. The results show that our weakly supervised model can detect anomalies immediately with no false alarms when anomalies occur. We also transfer and apply our pre-trained model to this well-known public dataset. Compared with previous work, our method significantly improves the recall rate by 20.71% and F1 by 10%. Thus we can confirm the feasibility of our framework and usability of inaccurate data. We also confirm that we can successfully detect all well-known anomalies in real-time and identify different anomaly categories of relationships when massive events occur.

The remainder of this paper is organized as follows. We first introduce related works (§ II) and present a comprehensive attack taxonomy (§ III). After that, we design the general framework in § IV, and introduce our dataset in § V. Then we prove the usability of our (weak) data and analyze our dataset in § VI. At last, we evaluate our method by testing the data collected from teacher systems and well-known anomalies (§ VII).

II. Related Works

The anomaly detection is applied to actively or passively detect BGP hijacking events then mitigate or inform target maintainers of AS. The vast majority of operators prefer to use passive anomaly detection systems, since they occupy the least amount of network bandwidth resources and are able to monitor anomalies more readily [13].

Among passive detection methods, control plane detection platforms are mostly used, including BGPmon [26], RIPE RIS [29] and RouteViews [30]. Furthermore, there are many ways to use these control platforms to achieve detection. For example, in ARTEMIS [17], the authors originally introduce to use real-time BGP updating information from BGP control platforms in order to achieve real-time detection. However, they typically use fixed rules which are not adaptive to new anomalies, *e.g.*, route leakage attacks or more than two hops hijacking. Testart et al. [18] use historical information from the control platform and the operator's mailing list to analyze the characteristics of serial hijackers and compare it with the legal AS from the Mutually Agreed Norms for Routing Security (MANRS) Association. Then the authors use random forest to capture the difference between serial hijackers and legitimate AS. However, their method is unable to capture the route hijacking and hence it can be easily bypassed by a sophisticated hijacker to avoid detection. Moreover, it fails to achieve real-time anomaly detection. Lu et al. [19] use BGPstream [5] as event data sets, which provide a method to get more data. However, their method requires one day of BGP characteristic information as input, which causes large detection delays. Furthermore, the approach does not implement route hijacking detection. Cho et al. [20] depend on AS hegemony [31] metric to detect the route hijacking. However, their method requires 15 minutes delays for getting AS hegemony scores from Internet Health Report (IHR) API [32]. Cheng et al. [23] use long short-term memory (LSTM) models [25] to identify worm attacks such as Nimda and Code Red II. However, the worm attacks are indirect attack [33] and they only use BGP volume features, making it impossible to capture other anomalies that have a small impact on the network (e.g., route hijacking). In their following work [24], they use a wavelet model combined with LSTM to realize anomaly detection and achieve better performance on route leakage. Nevertheless, they still do not provide a general framework or offer sufficient dataset to solve other anomalies.

In addition, [21] and [34] both use Support Vector Machine (SVM) to realize a framework for anomaly detection. However, in [21], the authors only consider the AS-volume feature, which is only suitable for worm attacks and interruptions. Their framework is unsuitable for prefix hijacks, fake route, or Defcon attacks. In contrast, [34] use more features and filter the features. Nevertheless, their model does not consider the relationship between abnormal timings, so it could not detect small influence attacks (*e.g.*, routing hijacking). They only analyze and verify in a small number of interruptions and worm events. Therefore, both of these tasks are lack of adaptation capabilities to deal with new abnormal events.

In recent years, [28], [35], [36] all propose new detection models. In [28], the authors adopt different graph features to detect BGP anomalies, but their method causes a great increase in complexity in feature extraction with little benefit.

[1]https://github.com/universetao/A-General-Framework-BGP-Anomaly-Detection.

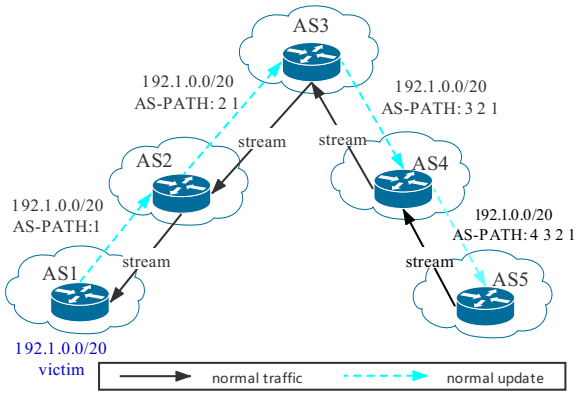


Fig. 1. Legitimate route. The paths announced by each AS are trustworthy and reliable

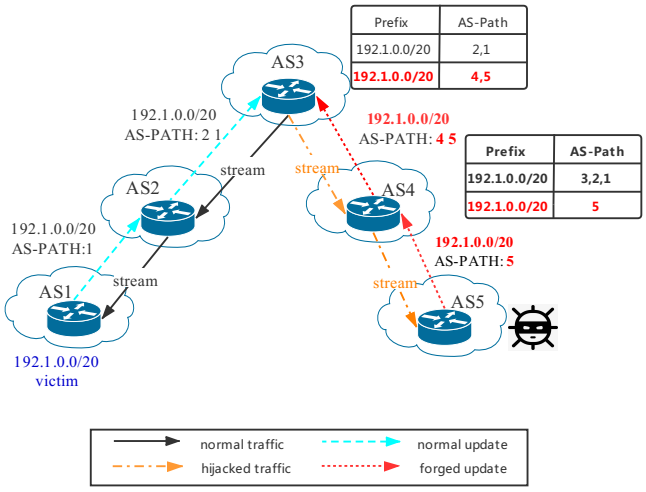


Fig. 2. Regular prefix hijacking. AS1 has the 192.1.0.0/20 address range. Hijacker AS5 illegally announces a route with the 192.1.0.0/20 address. According to the BGP shortest path principle, data from AS3 & AS4 to 192.1.0.0/20 is redirected to AS5

In [35], the authors use clustering techniques to raise alarms, but they do not provide experimental results. [36] put forward an autoencoder to support unsupervised learning, but they also indicate that their method performs poorly in route hijacking, because the hijacking event routes are similar to normal routes.

In conclusion, neither supervised nor unsupervised learning has been shown to successfully support BGP anomaly detection due to the high cost of data-labeling and the challenges of highly dynamic networks. In this paper, we propose using weakly supervised learning for BGP anomaly detection. Leveraging a self-adaptive weakly supervised learning model with a large and comprehensive real abnormal events data set, we provide a general framework that is able to detect all anomalies.

III. BGP Common Attack Taxonomy

BGP does not consider the authentication of autonomous systems in its design [37]. It assumes that announcements advertised by each autonomous system are trustworthy and reliable, and can be forwarded by peers. This design causes a series of BGP abnormalities, making the network prone to abnormal situations such as route leakage and hijacking. In addition, hijacking can be divided into prefix hijacking and routing hijacking. Based on the legitimate routes shown in Fig.1, we provide definitions and examples in the following subsections.

A. Prefix Hijacking

Prefix hijacking can be divided into two main categories: regular prefix hijacking and sub (more specific) prefix hijacking.

Regular prefix hijacking. As shown in Fig.2, the hijacker forges the target AS1's prefix in the Network Layer Reachability Information (NLRI). In this case, data from AS3 & AS4 to AS1 will be rerouted and stolen by AS5.

Sub-prefix hijacking. As shown in Fig.3, the hijacker not only forges the prefix in the NLRI announcement, but also

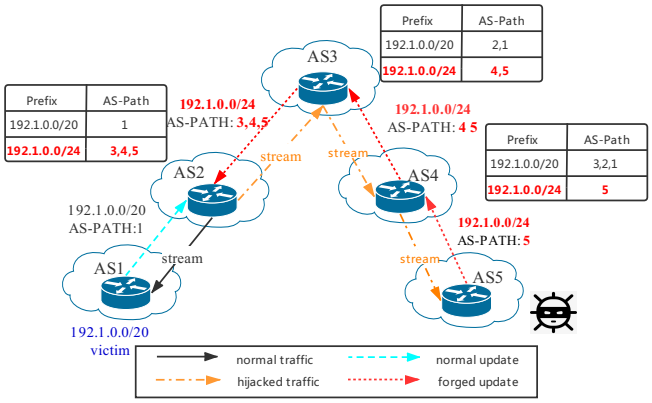


Fig. 3. Sub-prefix hijacking. AS5 declares a longer prefix than AS1. According to the longest match principle, all data flows to hijacker AS5.

makes the prefix longer than the target AS prefix, which causes the data to be rerouted to hijacker AS5.

Regular prefix hijacking generally attracts traffic near the hijacker according to the nearest principle. Moreover, sub-prefix hijacking will typically hijack more traffic because of the longest match principle.

B. Route Leakage

Route leakage is one of the most common BGP abnormal events. It may cause multiple types of issues, ranging from economic loss to network interruption. The definition of route leakage is that AS propagates a Provider's or Peer's route to another Provider or Peer. Based on the routing policy priority, the provider may choose a customer's route rather than the peer's or provider's route so that the network's routing will change drastically.

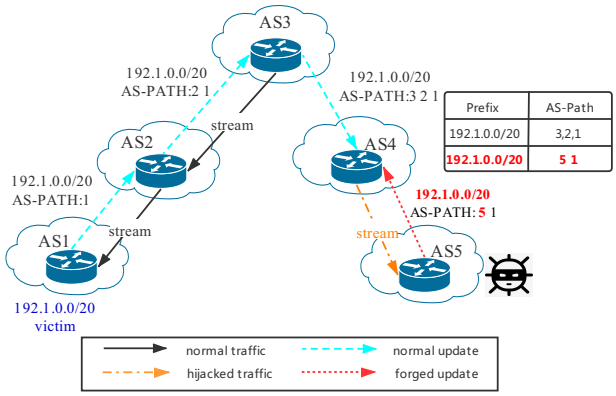


Fig. 4. Fake route attack. A hijacker forges a shorter route 5,1, which hijacks AS4's traffic.

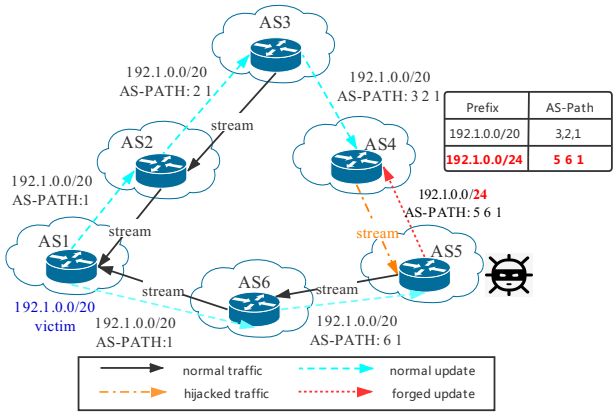


Fig. 5. Defcon attack. A hijacker forges a more specific prefix so that it can be used for MITM attacks.

C. Route Hijacking

In this attack scenario, the hijacker only needs to advertise a fake AS-PATH to the target AS. Suppose the fake path is shorter than the path from the compromised AS to the target AS. This will cause Man in the Middle (MITM) attack according to the AS-PATH shortest path priority principle. This attack is more difficult to detect because it avoids detection of multi-origin conflicts of origin prefixes (MOAS). According to the path authenticity, route hijacking can be divided into fake route (Fig.4) and Defcon attack (Fig.5). Compared with fake route, Defcon attack has a real path but forges a sub-prefix to realize MITM attacks. Moreover, Defcon is more challenging to detect because the path is reachable.

IV. Method

As shown in Fig.6, we propose a self-adaptive general framework based on weakly supervised learning. This is divided into the following modules: a knowledge collection module, a feature extraction module and an adaptive anomaly detection module. All the modules can be self-operated in ISP.

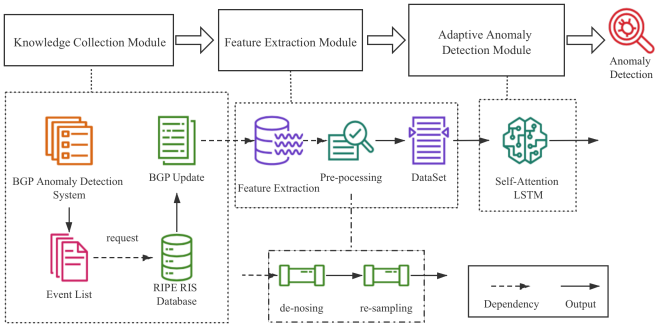


Fig. 6. A self-adaptive framework based on adaptive distillation learning using a knowledge collection module, a feature extraction module and an adaptive anomaly detection module.

The first module is designed to extract knowledge. It obtains event information from the well-known anomaly detection systems such as [5] and [8] and then obtains original BGP UPDATE messages from the RIPE RIS database. The second module is the feature extraction module that extracts features from the original BGP UPDATE messages. The last module is the adaptive anomaly detection module, which uses the LSTM model based on a self-attention mechanism. The self-attention mechanism is used to explore the internal relationships of abnormal event time series. In addition, the hidden layer of LSTM adaptively assigns weights to features, thereby supporting the selection of features. These modules are described in detail in the following subsections.

A. Knowledge Collection Module

The knowledge collection module collects abnormal events by accessing well-known anomaly detection systems. It can be inaccurate data where the given labels are not always based on the ground truth. Nevertheless, the data follows the central limit theorem, which is proved in section VI. Moreover, the different distribution between anomalies can be learned by our tailored weakly supervised learning model. Each abnormal event includes the corresponding information in the following Table I. Note that not all the events have these two attributes: event end time and hijacker AS. For example, when a prefix hijacking happens, the anomaly may last for a long time, so there would be no end time attribute. In most cases, there is no specific hijacker AS for the breakout event.

Then, according to the list of abnormal events, we use the pybgpstream python API [38] to obtain original BGP UPDATE packets from the control platform RIPE RIS. After this module finish, we input original BGP UPDATE packets as a traffic to the next module.

B. Feature Extraction Module

In this module, we extract features from the original BGP traffic data. We define the BGP traffic data as a time sequence given as $S_n = (x_1, x_2, x_3, ..., x_n)$, collected in n time points with the interval based on per minute. Each x_t represents m-dimension features in the t^{th} time interval given as $x_t =$

TABLE I
EVENT ATTRIBUTE

Event Attribute	Necessity
Abnormal event type	⋆
Victim AS	⋆
Event start time	⋆
Event end time	
Hijacker AS	

$(f_1, f_2, f_3, ..., f_m)$. In the following subsections, we further demonstrate how we handle this timing data and features.

1) Sliding Window: We adopt a sliding time window to capture time dimension changes with size defined as w. It is difficult to find dynamic changes in the sequence without a sliding window. For example, abnormal events often exhibit a large number of withdrawal announcements in the first few minutes and a large number of NLRI announcements in the next few minutes. Therefore, it is necessary to use a sliding time window to mine abnormal feature changes and time relationships. Thus, the data at the j^{th} time can be expressed as $X_j = (x_{j-w+1}, x_{j-w+2}, ..., x_j)$.

2) Feature Set: As shown in the table II, we divide the features into four categories: prefix features, peer features, AS-volume features and AS-PATH features. We discuss these in the following subsections.

Prefix Features: The features f_{1-4} characterize prefix hijacking events. f_4 can also identify MITM attacks.

Peer Features: Features f_{5-7} are related to the number of peers. f_5 and f_7 describe the target AS notification range and f_6 reflects MITM attacks.

AS Volume Features: Features f_{8-13} are volume-type features. To further distinguish route engineering from hijacking, we separate announcements into MOAS announcements and other normal announcements.

AS-PATH Features: We can estimate the scope of the target AS update by measuring the AS-PATH length in a given period. The AS-PATH feature is mainly used to describe the route changes. We use the edit distance algorithm to estimate the path difference. The edit distance algorithm is given in Formula 1.

$$d(i,j) = \begin{cases} d(i-1, j-1), & i = j \\ \min(d(i-1, j-1), d(i-1, j), d(i, j-1)) + 1, & i \neq j \end{cases} \quad (1)$$

where $d(i, j)$ represents the least times changed from AS-PATH l_1 (length i) to AS-PATH l_2 (length j). When $i = j$, $d(i, j)$ obviously equals the previous $d(i-1, j-1)$. In contrast, $d(i-1, j-1)+1$ means that the i-th hop changes to the j-th hop. $d(i-1, j)+1$ means that the l_2 sequence adds a route at j. $d(i, j-1)+1$ means that the l_2 sequence has one less route to j. The algorithm complexity is $O(l_1 l_2)$.

However, knowing only the edit distance but not whether the path length increases or decreases makes it challenging to distinguish hijacking incident from the network engineering migration. Therefore, we record the increase and decrease

TABLE II
ALL FEATURE SET

ID	type	Name	Description
1	Prefix	MOAS_prefix	The number of prefix's origin conflicting with other AS
2		MOAS	Number of AS that conflict with the target AS
3		new_MOAS	Number of new prefixes announced by other AS
4		new_prefix	Number of new sub-prefixes belong to target AS
5	Peer	peer_num	Number of peers
6		peer_increase	Number of new peers
7		MPL	Maximum path
8	AS Volume	MOAS_Ann	Number of announcements from origin conflict AS
9		Own_Ann	Number of announcements from target AS
10		Dup_ann	Duplicate announcements with same AS-PATH
11		Unique_WD	The unique number of withdrawn prefixes
12		Withdraw	Number of withdrawals
13		Diff_Ann	Number of new paths announced after withdrawing an old path
14	Feature Set	AS-Path	PL_{it} AS-PATH length set
15			$Diff_{jt}$ AS-PATH edit distance set
16			Ann_longer_{lt} AS-PATH length increase set
17			$Ann\_shorter_{kt}$ AS-PATH length decrease set
18	Sum	pl_sum	Sum of the all path lengths
19		sum_diff	Sum of edit distances
20		sum_ann_longer	Sum of the distances increases
21		sum_ann_shorter	Sum of the distance decrease
22	Average	avg_pl	Average length of AS-PATH
23		avg_diff	Average edit distance
24		avg_longer	Average increase distance
25		avg_shorter	Average decrease distance

for each path update. We define the Features sets f_{14-17} as follows:

$$\mathrm{PL_{it}},\ \mathrm{Diff_{jt}},\ \mathrm{Ann\_longer_{lt}},\ \mathrm{Ann\_shorter_{kt}}, \quad (2)$$

where these feature denote the number of BGP announcements with path length i, edit distance number j, path increase number l and path decrease number k at the t^{th} time, respectively.

Each feature in f_{18-21} is used to describe the network's global situation of target AS based on the sum of AS-PATH features. For example, when a breakout occurs, the sum of the path increases will be unusually large. In addition, features f_{22-25} are based on f_{18-21} divided by the number of peers. This is used to obtain each peer's degree of change.

C. Adaptive Anomaly Detection Module

As shown in Fig.7, the model consists of a Batch Normalization layer (BN) [39], an LSTM model [25], a self-attention layer [40] and a fully connected layer. We use the processed BGP traffic $X_j = (x_{j-w+1}, x_{j-w+2}, ..., x_j)$ as input. First, the BN layer normalizes the X_j to $BN(X_j)$. Then, we send $BN(X_j)$ into the LSTM and output representation sequences $X'_j = (x'_{j-w+1}, x'_{j-w+2}, ..., x'_j)$. Then, we put the

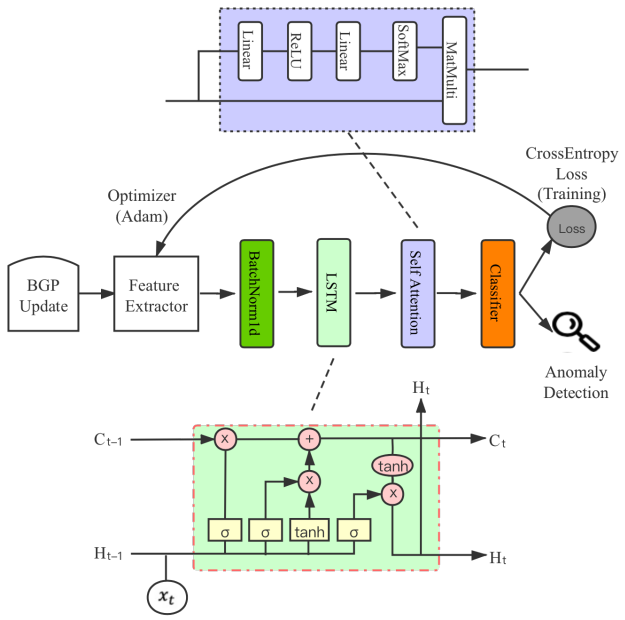


Fig. 7. Adaptive anomaly detection model. In the LSTM model, x_t represents the input set at the t^{th} time. C_t is the cell state at the t^{th} time, and H_t is the state of the output at the t^{th} time. σ represents the neural network layer with sigmoid activation function, and $tanh$ represents the $tanh$ activation function. $\oplus and \otimes$ represent the pointwise operator of adding and multiplication respectively.

representation sequences X'_j into the self-attention to mine the importance distribution $\omega = (\omega_{j-w+1}, \omega_{j-w+2}, ..., \omega_j)$ from the global perspective and multiply X'_j by ω. Finally we send these redistributed representation sequences into the fully connected layer and the Softmax layer to get the probability of each anomaly.

We design the function of the adaptive LSTM to transfer the information extracted from each timestamp to the next timestamp and identify the anomaly at the last timestamp. In the LSTM model, the cell state is throughout of the entire sliding time window. Simultaneously, through logical gates, the new input x_t is selectively added to the cell state, and the past information is selectively removed. Based on these methods, we can capture long-term information from the entire time window. The LSTM's neural network layer can adaptively compute the importance of each feature. The self-attention mechanism generates a series of weights and then uses these weights to multiply the input parameters. It then uses backpropagation of loss to realize the redistribution of the input parameters. The advantage of self-attention is to observe the importance of each timestamp information from a global perspective, then to weight the input to the output.

V. Dataset

A. Data Collection

Weak dataset collection: We use the most famous BGP anomaly detection systems (*e.g.*, BGPStream [5] and BGPObservatory [8]) as the teacher model for abnormal events.

TABLE III
Event Collection Sources

	BGPStream	BGP Hijacking Observatory	MANRS
Prefix Hijack	√ (535)	√	×
Route Leak	√ (290)	×	×
Breakout	√ (1090)	×	×
Fake Route	×	√(532)	×
Defcon	×	√(638)	×
Normal	×	×	√ (21)

TABLE IV
Well-known Anomalies

Anomalies	Anomaly Start Date	Duration(min)
TTNet(AS 9121)	Dec 24,2004(9:20 UTC)	627
IndoSat(AS 4761)	April 2,2014(18:25 UTC)	150
TM(AS 4788)	June 12,2015(8:43 UTC)	182
AWS(AS 200759)	April 22, 2016(17:10 UTC)	115

In addition, we use ASes of the MANRS initiative [41] as the teacher model for normal events, because these ASes deploy the RPKI and can be trust relatively. After the data preprocessing in § V.B, we collect 3085 abnormal events, which are split into 81% for training data, 9% for evaluation data and 10% for testing data. The given labels are not always ground truth in the events though. The detection range of each anomaly detection system and each event number are shown in Table III. For training and testing on the weak dataset, each abnormal event acquires 20 minutes before and after the anomaly for an abnormal data set. At the same time, we obtain 20 representative ASes from MANRS as providers of normal data and collect 1440 minutes of data for each AS.

Ground truth dataset collection: In order to explain the impact of inaccurate data on the model, we use well-known anomalies [24] to test our trained model. This uses inaccurate testing data for evaluation. The dataset contains four well-known BGP events shown in Table IV. These include data from Turkish Telecom (TTnet) [3], Indosat (Indonesia) [42], Telecom Malaysia (TM) [2], and the attack on Amazon Web Services (AWS) [4]. We use pybgpstream [38] to extract historical BGP UPDATE packets from the RIPE RIS [29] where each case contains 720 minutes of data before and 720 minutes after the event occurrence. Furthermore, we label the data based on the duration of the anomaly.

B. Data Pre-processing

Drop duplicate events and low coverage events: Since the teacher anomaly detection systems may report the duplicate events within a period, which may hide the true event start time. Hence, we drop the duplicate events with an interval of 15 minutes and only keep the first event. In addition, our BGP data is collected from RIPE RIS collectors, which may exist some regions with poor coverage. Thus, we drop events whose f_7 sum is less than 5 in the entire collection period.

Dataset de-noising: To prevent features from changing drastically during initialization, we use the first 100 minutes

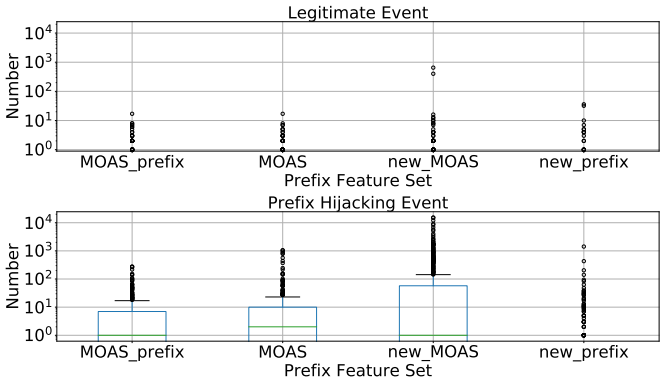


Fig. 8. Comparing prefix hijacking events with normal events on prefix feature sets. From left to right, f_{1-3} significantly distinguish prefix hijacking events from normal events, whereas f_4 is used to characterize route hijacking (*e.g.*, Defcon). Note that the Y-axis is based on a log scale.

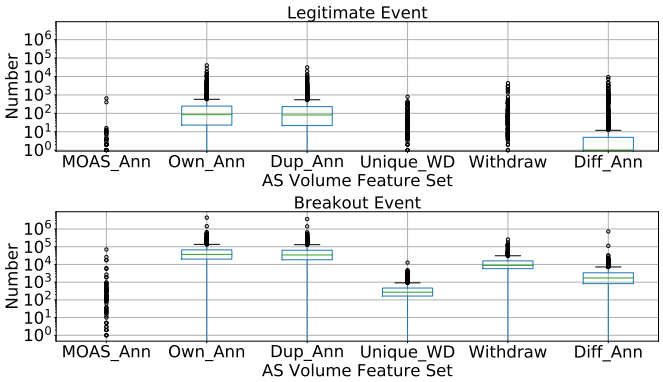


Fig. 9. Comparing breakout events with normal events based on AS volume sets. From left to right, breakout anomalies are 1000 times higher than ordinary events based on f_{9-13}. f_8 is used to characterize prefix hijacking. Note that the Y-axis is based on a log scale.

as an initialization phase. Then we discard it from the training data.

dynamic stride sampling method: To address class imbalances, we design a sampling method that uses different stride sizes for normal data and abnormal data in the training phase. Based on the following formula 3 and 4, we can get the size of the data set.

$$D_i = \frac{E_i - W_c}{s_c} + 1 \tag{3}$$

$$T_c = \sum_{n=1}^{N_c} D_n \tag{4}$$

where D_i represents the data size provided by an event i, E_i represents the time in minutes of the event i. W_c represents each category's sliding window size, and s_c defines each category's stride. N_c represents the event number of each category, and T_c defines the total data size of each category.

Data Pre-processing for analysis: To facilitate the analysis of the features' distribution, we use the sum of the features based on 15 minutes before and after the anomaly as a single sample, i.e., a total of 30 minutes for each anomaly. For normal events, we use a 30 minute sliding window to generate normal event sets.

VI. Data Analysis

In this section, we show the usability of our weak dataset and analyze it. The results demonstrate that different anomaly categories are indeed separable, and the dataset satisfies the central limit theorem.

A. Prefix Features Data Analysis

As shown in Fig.8, we observe that prefix hijacking is significantly different from normal events in the prefix feature set. As expected in § IV, f_{1-3} can identify prefix hijacking well, while f_4 is used to distinguish routing hijacking (*e.g.*, Defcon). In detail, only a few normal events' f_{1-3} are more than 1, but most of prefix hijacking events' f_{1-3} exceed 1000.

B. AS Volume Features Data Analysis

As shown in Fig.9, we see that the AS volume f_{9-13} distinguish breakout events from normal events. In particular, breakout anomalies are 1000 times higher than normal events based on these features, and breakout anomalies have no small values. The reason is that when a breakout event occurs, there will be a large number of unreachable routes, resulting in a large number of routing updates and withdrawals. In addition, f_8 is used to identify prefix hijacking events.

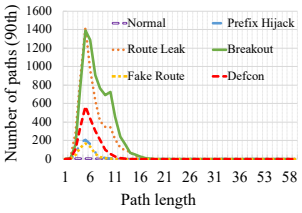
C. AS-PATH Features Data Analysis

We show the distribution of path feature sets based on different anomalies. Fig.10 (a) shows the 90th quantile distribution of the path length. We observe that different anomalies have their own pattern on the update path range, and normal events have little change. In detail, breakout and route leakage events cause a massive change on the AS-Path, whereas Defcon hijacking has a medium influence on AS-Path. Additionally, prefix hijacking and fake route hijacking have less change on AS-Path. For those shown in Fig.10 (b) (c) (d), we can also easily distinguish them from normal events.

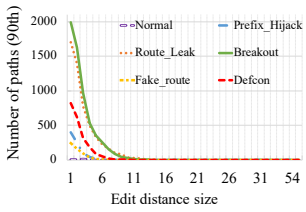
We also find an interesting phenomenon. The AS-Path features all have a heavy tail, and the total proportion of the tail is less than 1% of the sum of the respective path features. This shows an apparent abnormal behavior when the features' values are too large, so the information in the heavy tail needs to be maintained. Specifically, we find that the heavy tail part of the path length is greater than 21, and the heavy tail part of the edit distance is greater than 14, whilst the heavy tail part of the increase and decrease distance are both greater than 11. Hence, we divide features sets f_{14-17} into two groups according to these thresholds shown in Table V. This approach has the advantage that the features are greatly reduced while the data information captured by the features are maintained.

VII. Experimental Results

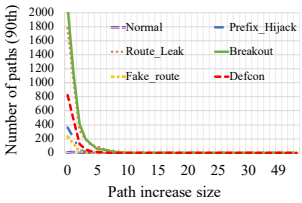
In this section, we first conduct hyperparameter selection experiments, and then we show the interpretability of our



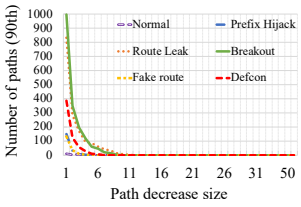
(a) 90th quantile distribution plot of path length



(b) 90th quantile distribution plot of edit distance



(c) 90th quantile distribution plot of increase path length



(d) 90th quantile distribution plot of decrease path length

Fig. 10. Heavy tail distribution of path features, where each proportion of heavy tail is less than 1% sum of the respective path feature. In addition, compared with normal events, the anomalies all exhibit an abnormal behavior

TABLE V
AS PATH ADDING FEATURE

ID	Name	Description
26	PL_{1-21}	AS-PATH length of 1-21
27	$Diff_{1-14}$	Edit distance 1-14
28	Ann_longer_{0-11}	Path increases by 0-11 number of paths
29	$Ann\_shorter_{0-11}$	Path decreases by 0-11 number of paths
30	PL_sum_{above}	The sum of AS-PATH length over 21
31	$Diff\_sum_{above}$	The sum of edit distances over 14
32	$Longer\_sum_{above}$	The sum of increase distances over 11
33	$shorter\_sum_{above}$	The sum of decrease distances over 11

model. We then evaluate our method by testing the data collected from the teacher anomaly detection and other well-known anomalies. The results confirm the feasibility of our weak supervision-based approach.

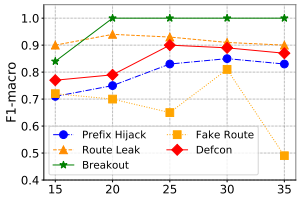
A. Evaluation Metrics

We divide the anomaly detection into multiple binary classification problems, where a positive sample denotes anomaly. Then we use standard metrics [43] for binary classification, which include Anomaly's Precision (PR), Recall (RC), F-measure (F1), and these macro metric (e.g., PR-macro, F1-macro). Macro is a metric that appropriately represents unbalanced classes.

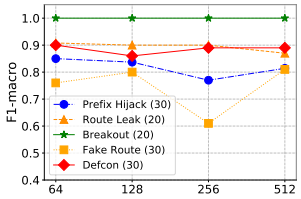
B. Hyperparameter Selection

The sliding time window has a trade-off between length and performance. If the sliding time window is too large or too small, it will affect the model's recognition of the time series. Therefore, we conduct experiments on the sliding window size for each anomaly category.

The results are shown in Fig.11 (a). We find that all anomalies reach the highest performance at 20-30, because



(a) Sliding Window Size



(b) LSTM Hidden Size

Fig. 11. The F1 with different hyperparameters.

these sliding window sizes are neither too small to capture the time series changes nor too large to weaken the difference between abnormal and normal features. Breakout and route leak anomalies reach the highest performance and remain stable since we have a sliding time window of 20. Defcon reaches the highest performance at 25, whereas the prefix hijack and fake route reach the highest performance at 30. In addition, we observe that the fake route drops sharply in time windows of 35. This is because the fake route attack has a small impact area, and if the sliding time window size increases, the abnormal data tends to be more similar to normal routing changes. We find that route hijacking requires a greater sliding window size to identify anomalies and hence they are more challenging to recognize.

Then we show the influence of different LSTM hidden sizes on the recognition of anomalies. Note that in Fig.11 (b), comparing route hijacking with route leakage or breakout anomalies, we can find that changing hidden size has less impact on the route leakage or breakout anomalies, whereas hidden size changes will lead to larger performance changes for route hijacking models. Moreover, for breakout and route leaks, 64 hidden sizes are enough to achieve a good detection result. However, for route hijacking, we need a larger hidden size to detect anomalies. Besides, we find a drop in the fake route model with 256 hidden sizes. The reason is possibly the over-fitting caused by the high epoch, and the fake route model is more sensitive to variation of hidden sizes. Therefore, we suggest setting 128 hidden sizes with a lower epoch.

C. Interpretability Analysis of Self Attention

By visualizing each anomaly category's self-attention weight, we can find that the attention model has a different emphasis based on different types of anomalies. As shown in Fig.12, for prefix hijacking, the weights increase with the time series, which is more reasonable than the traditional model using the same weights for all timestamps. In contrast, the detection model of route leak pays more attention to the historical information, because the changes in routing policy are more difficult to be detected. In addition, the large-scale events (*e.g.*, breakout) will cause a large number of messages to be unreachable, which can have a huge impact on the network. Thus, the attention model needs to focus more on the features of the most recent timestamp. However, for typical path hijacking (*e.g.*, fake route and Defcon), unlike prefix hijacking and breakout, path hijacking is difficult to detect

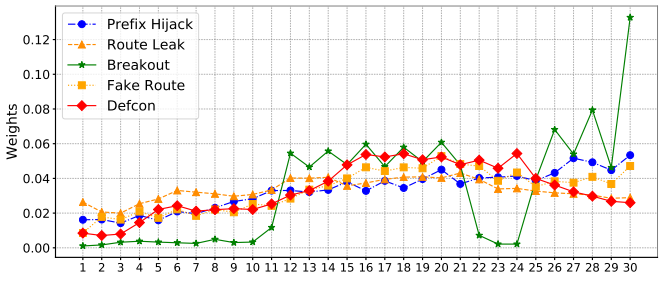


Fig. 12. Attention weights of each event.

because of a smaller scope of influence. Hence, the model needs to pay more attention to the past information too.

D. Comparison of Different Models and Feature Sets

Baseline model parameters setting: For all model training and testing, we use a sliding window size of 30. For Random Forest and SVM models, we reshape two-dimensions (sliding window size, feature size) into one dimension. We use grid search and k-fold cross-validation for the baseline models to select the best parameters and confirm that all baseline models achieve their best performance. All experimental results in this subsection are shown in Table VI.

Adaptive Feature Selection: The related frameworks only use part of our features (*e.g.*, prefix features, AS volume type with peer features or AS volumes only) to identify anomalies, hence we combine them and set them as baseline features (BL). According to the experiments, if we only use baseline features, the Random Forest model's performance will drop significantly. Compared with Random Forest, SVM's robustness is slightly better. In other words, except for breakout or route leak attacks, SVM on the other attacks' F1-macro only decrease by 3-12%. The results also show that the baseline feature sets are suitable for large-scale anomalies such as breakout attacks. Therefore, we need to add more path-related features to capture other small-scale anomalies. We also observe that the baseline features are equivalent to selected features from all features (All) for breakout anomaly events, which improves the SVM model's detection in breakout events. Hence, it is worth noting that our model achieves the best results with all features, implying that the model has the ability to mine the importance of features.

Abnormal Adaptability: Compared with baseline models under all feature sets, the self-attention LSTM model has better generality and adaptability to detect various abnormal types, and it obtains the highest precision and F1 values for most abnormalities. Furthermore, our work significantly increases the average of the F1 macro, i.e., it has a high recall and accuracy rate. As for the Random Forest model, due to its own feature filtering capability, it obtains the highest F1-macro for route leaks, but it does not consider changes in timing. Hence, compared to our model, the performance of Random forest on other anomalies is far behind. As for the SVM model, its PR-macro is similar to our model, but it is far inferior to the F1-marco, i.e., the recall rate is weaker.

TABLE VI
PERFORMANCE OF EACH MODEL AND FEATURE SET

	metric	Random Forest		SVM		Ours
		All	BL	All	BL	All
Prefix Hijack	PR-macro	65	64	**89**	**89**	**89**
	F1-macro	63	61	82	74	**86**
Route Leak	PR-macro	97	58	**98**	94	97
	F1-macro	**96**	55	95	**96**	93
Breakout	PR-macro	91	82	98	**100**	**100**
	F1-macro	91	78	95	**100**	**100**
Fake Route	PR-macro	73	64	**88**	85	84
	F1-macro	74	61	73	64	**82**
Defcon	PR-macro	80	70	**94**	93	**94**
	F1-macro	82	67	87	84	**92**
Average	F1-macro	81.2	64.8	86.4	83.6	**90.6**

Comprehensive and Self-operated: In this experiment, we successfully distill the knowledge from the anomaly detection system and achieve better performance through our framework. Moreover, our model has more comprehensive anomaly detective ability than any detection system and can be self-operated in target AS.

E. Testing on Public Well-known Anomalies

To better show that the models trained on the weak dataset have better generalization performance, we use a well-known anomaly dataset [28] to test the trained model.

Immediate and zero false alarm: As shown in Fig.13, we visualize the recognition of events with different anomaly models. Our experiments show that anomalies can be identified immediately. We also find that after an anomaly occurs, rerouting in the network will continue for a period of time. Therefore, it is safe to assume that only the warning generated before an anomaly occurs is a false alarm. It is also worth noting that our false alarm rate is 0. In contrast to our model, only adopting the LSTM model will cause a delay for about four minutes in detecting the AS4788 anomaly. We believe this is caused by the lack of attention mechanism, which learns the importance of time steps and helps our detect model become more quickly and effectively.

Different abnormal categories' relationship: For the TTNet (AS9121) and IndoSat (AS4761) events, we can observe that the interruptions caused by route leaks can be accurately identified. We can also find that the routing will be re-routed immediately after the interruption with a large number of new routing paths, which are identified as routing hijacking by our model. Besides, for TM (AS 4788) and AWS (AS 200759), although routing leaks occur, they do not lead to network interruptions, indicating that these two networks are more robust to the anomalies. We discover the relationship between abnormal categories when massive abnormal events occur through such analyses. Meanwhile, this highlights the generalization of our method on other datasets.

F. Transfer Learning on Well-known Anomalies

In order to achieve better performance, we combine the existing ground truth dataset. We use the models learning from

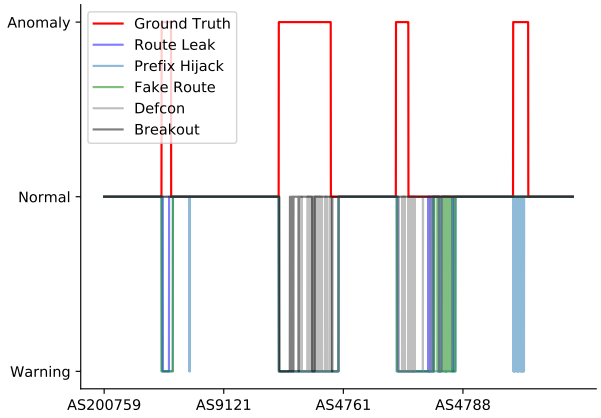


Fig. 13. Anomaly detection based on well-known anomalies. Each segment of the X-axis represents an abnormal event. Above the Normal line is the ground truth, and below the Normal line is the prediction alert of our model. We use different categories of models to detect these events, to observing the network's continuous impact and to show the relationship between abnormal categories when massive abnormal events occur.

the weak dataset as a pre-trained base model, and transfer our models to well-known anomalies. We transfer the pre-trained models to a small sample of abnormal data for fine-tuning, in which the last layer and backbone network have different small learning rates. Furthermore, we use the models as an ensemble model to detect anomalies. Thus, the model can adapt to different sizes of AS by setting voting thresholds.

Similar to [28], we split the re-trained data into training and testing dataset for different incident combinations shown in Table VII. This operation allows the models to infer their performance on the testing dataset that does not influence the training dataset itself.

Our baseline model is the state-of-the-art BGP detection model MLGF [28]. MLGF uses graph features with the SVM. The graph features are derived from the AS topology, which include node centrality [44], clique theory [45], clustering coefficient [46] and eccentricity [47].

We hereby compare our ensemble model with the MLRF model, which only trains on ground truth data. As shown in Table VIII, our method has a significant improvement both in RC and F1 metrics. Specifically, our model improves the recall by 20% when the PR is only reduced by 1.34%. Furthermore, our method can detect anomalies more rapidly, because our

TABLE VII
OUR CLASSIFIER SET-UP

Scheme	Training				Test
	TTNet	IndoSat	TM	AWS	
Scheme A	×	√	√	√	TTNet(AS 9121)
Scheme B	√	×	√	√	IndoSat(AS 4761)
Scheme C	√	√	×	√	TM(AS 4788)
Scheme D	√	√	√	×	AWS(AS 200759)

TABLE VIII
PERFORMANCE OF TRANSFER LEARNING MODELS

Scheme	MLGF			Ours		
	PR	RC	F1	PR	RC	F1
Scheme A	**89**	82	85	87	**100**	**93**
Scheme B	**97**	93	**95**	87	**100**	93
Scheme C	**100**	76	86	**100**	**87**	**93**
Scheme D	93	58	72	**100**	**86**	**93**
Average	**94.75**	77.25	84.5	93.5	**93.25**	**93**
Comparison				**↓1.34%**	**↑20.71%**	**↑10.06%**

method is based on a one-minute time bin, whereas their method is based on five-minutes. It is also noteworthy that the high recall rate is very important since when the anomaly is ignored, which may cause irreparable economic loss or safety problems to the enterprise/country. In addition, our ensemble model has a good generalization performance and can be transferred to various AS as an early warning system. Specifically, when an anomaly occurs, different measures can be adopted according to the strength of the alarm confidence. For example, in the case of a strong confidence alarm, the spread of abnormal BGP updates can be directly prevented. For weak-confidence alarms, we can implement Route Flap Dampening or limit the rate of update or use human/expert review.

The results show that our weak dataset solves the problem of insufficient BGP data and can bring more priori knowledge. Besides, the pre-trained model can also be transferred to other ground truth datasets to achieve better performance.

VIII. CONCLUSION

With the emergence of more BGP anomalies, neither supervised learning nor unsupervised learning can implement BGP anomaly detection due to the high cost of the data-labeling processes and the dynamic nature of the network. Therefore, in our work, we propose a weakly supervised learning-based framework to deal with anomalies. We distill the knowledge from BGP from third-party systems. We then use the weak dataset to train a self-attention LSTM model, which can tackle data noise. Furthermore, we transfer the pre-trained models to the target AS dataset for further fine-tuning. As far as we are aware, this is the first work that uses weak supervision for BGP anomaly detection. The proposed framework is self-operated and can avoid cumbersome update operations on third-party anomaly detection systems. We test the framework on well-known anomalies and achieve high performance.

ACKNOWLEDGMENT

This work is supported in part by the National Key Research and Development Program of China under Grant 2018YFB1800204, National Natural Science Foundation of China under grant No. 61972189 and the Shenzhen Key Lab of Software Defined Networking under grant No. ZDSYS20140509172959989.

REFERENCES

[1] K. on Security, "Notorious 'hijack factory' shunned from web," https://krebsonsecurity.com/tag/bitcanal/.

[2] A. Toonk, "Massive route leak causes internet slowdown," *Technical Report*, 2015.

[3] A. C. Popescu, B. J. Premore, and T. Underwood, "Anatomy of a leak: As9121," *Technical Report*, 2005.

[4] A. Toonk, "Large hijack affects reachability of high traffic destinations," *Technical Report*, 2016.

[5] "BGPStream," https://www.bgpstream.com.

[6] M. Chen, M. Xu, Q. Li, and Y. Yang, "Measurement of large-scale BGP events: Definition, detection, and analysis," *Computer Networks*, vol. 110, pp. 31–45, 2016.

[7] M. Chen, M. Xu, Y. Yang, and Q. Li, "A measurement study on the distribution disparity of BGP instabilities," in *2016 IEEE 41st Conference on Local Computer Networks (LCN)*, 2016, pp. 19–27.

[8] "BGP observatory," https://dev.hicube.caida.org/feeds/hijacks/.

[9] M. Prince, "August 30th 2020: Analysis of centurylink/level(3) outage," https://blog.cloudflare.com/analysis-of-todays-centurylink-level-3-outage/.

[10] H. Birge-Lee, L. Wang, J. Rexford, and P. Mittal, "SICO: Surgical interception attacks by manipulating BGP communities," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2019, pp. 431–448.

[11] R. Bush and R. Austein, "The resource public key infrastructure (RPKI) to router protocol," 2013.

[12] S. Kent, C. Lynn, and K. Seo, "Secure border gateway protocol (S-BGP)," *IEEE Journal on Selected areas in Communications*, vol. 18, no. 4, pp. 582–592, 2000.

[13] P. Sermpezis, V. Kotronis, A. Dainotti, and X. Dimitropoulos, "A survey among network operators on BGP prefix hijacking," *ACM SIGCOMM Computer Communication Review*, vol. 48, no. 1, pp. 64–69, 2018.

[14] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang, "PHAS: A prefix hijack alert system." in *Proceedings of the 2006 USENIX Security symposium*. USENIX, 2006, pp. 1–3.

[15] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis, "A light-weight distributed scheme for detecting ip prefix hijacks in real-time," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4, pp. 277–288, 2007.

[16] Z. Zhang, Y. Zhang, Y. C. Hu, Z. M. Mao, and R. Bush, "ISPY: Detecting ip prefix hijacking on my own," in *Proceedings of the 2008 ACM SIGCOMM conference on Data communication*. ACM, 2008, pp. 327–338.

[17] P. Sermpezis, V. Kotronis, P. Gigis, X. Dimitropoulos, D. Cicalese, A. King, and A. Dainotti, "ARTEMIS: Neutralizing bgp hijacking within a minute," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2471–2486, 2018.

[18] C. Testart, P. Richter, A. King, A. Dainotti, and D. Clark, "Profiling BGP serial hijackers: Capturing persistent misbehavior in the global touting table," in *Proceedings of the Internet Measurement Conference*, 2019, pp. 420–434.

[19] Y. Lu, "Research and implementation of BGP abnormal event detection based on machine learning," Master's thesis, Beijing University of Posts and Telecommunications, 2018.

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

[21] J. Li, D. Dou, Z. Wu, S. Kim, and V. Agarwal, "An internet routing forensics framework for discovering rules of abnormal BGP events," *ACM SIGCOMM Computer Communication Review*, vol. 35, no. 5, pp. 55–66, 2005.

[22] M. Karimi, A. Jahanshahi, A. Mazloumi, and H. Z. Sabzi, "Border gateway protocol anomaly detection using neural network," in *Proceedings of 2019 IEEE International Conference on Big Data*. IEEE, 2019, pp. 6092–6094.

[23] M. Cheng, Q. Xu, J. Lv, W. Liu, Q. Li, and J. Wang, "MS-LSTM: A multi-scale LSTM model for BGP anomaly detection," in *Proceedings of the 2016 IEEE 24th International Conference on Network Protocols*. IEEE, 2016, pp. 1–6.

[24] 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.

[25] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[26] "BGPmon(colorado State University," http://www.bgpmon.io/.

[27] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National Science Review*, vol. 5, no. 1, pp. 44–53, 2017.

[28] 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*. ACM, 2019, pp. 35–41.

[29] "RIPE network coordination center (NCC). routing information service (RIS)," http://www.ripe.net/data-tools/stats/ris/routing-information-service/.

[30] "The Route Views Project," http://www.routeviews.org/.

[31] R. Fontugne, A. Shah, and E. Aben, "The (thin) bridges of as connectivity: Measuring dependency using as hegemony," in *International Conference on Passive and Active Network Measurement*. Springer, 2018, pp. 216–227.

[32] I. R. Lab, "Internet health report," https://ihr.iijlab.net/.

[33] 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, 2016.

[34] A. Allahdadi, R. Morla, and R. Prior, "A framework for BGP abnormal events detection," *arXiv preprint arXiv:1708.03453*, pp. 1–8, 2017.

[35] A. Putina, S. Barth, A. Bifet, D. Pletcher, C. Precup, P. Nivaggioli, and D. Rossi, "Unsupervised real-time detection of BGP anomalies leveraging high-rate and fine-grained telemetry data," in *Proceedings of the 2018 IEEE INFOCOM Conference on Computer Communications Workshops*. IEEE, 2018, pp. 1–2.

[36] K. McGlynn, H. Acharya, and M. Kwon, "Detecting BGP route anomalies with deep learning," in *Proceedings of 2019 IEEE INFOCOM Conference on Computer Communications Workshops*. IEEE, 2019, pp. 1039–1040.

[37] R. Lychev, M. Schapira, and S. Goldberg, "Rethinking security for internet routing," *Commun. ACM*, vol. 59, no. 10, p. 48–57, 2016.

[38] C. Orsini, A. King, D. Giordano, V. Giotsas, and A. Dainotti, "BGPStream: A software framework for live and historical BGP data analysis," in *Proceedings of the 2016 Internet Measurement Conference*. ACM, 2016, pp. 429–444.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*. PMLR, 2015, pp. 448–456.

[40] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *Proceedings of the 2015 International Conference on Learning Representations*, 2015, pp. 1–15.

[41] "Mutually agreed norms for routing security(MANRS)," https://www.manrs.org/.

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

[43] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[44] 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.

[45] J. Orlin, "Contentment in graph theory: covering graphs with cliques," in *Indagationes Mathematicae*. Elsevier, 1977, pp. 406–424.

[46] J. Saramäki, M. Kivelä, J.-P. Onnela, K. Kaski, and J. Kertesz, "Generalizations of the clustering coefficient to weighted complex networks," *Physical Review E*, vol. 75, no. 2, p. 027105, 2007.

[47] J. M. Hernández and P. Van Mieghem, "Classification of graph metrics," *Delft University of Technology: Mekelweg, The Netherlands*, pp. 1–20, 2011.