Recognizing BGP Communities Based on Graph Neural Network

Yuntian Tan, Wenfeng Huang, Yang You, Shen Su, and Hui Lu

Abstract

The identification of BGP community attributes holds significant importance in routing modeling for understanding their impact on routing behavior and policies. However, when it comes to identifying BGP community attributes, existing rule-based solutions suffer from low automation and coarse granularity for relying on human knowledge, and the immutability of rule-based solutions makes them impractical for time-sensitive applications. Consequently, our research proposes a consistent and automated approach for identifying BGP community attributes in AS-level. We pick out the related autonomous system (AS) paths and community attribute numeric identifiers within BGP forwarding entries. Using these AS paths and identifiers we construct a topological graph structure and generate relevant feature embeddings for nodes and edges. Subsequently, we build a graph neural network (GNN) model consisting of a residual network for convolution and a fully connected layer, which can preserve and highlight the differences in the features of different communities. We employ this model to classify BGP communities. It turns out that the accuracy of our design can surpass 96%. According to the experimental result, our approach outperforms other state-of-the-art methods.

Introduction

BGP, which stands for Border Gateway Protocol, is a routing protocol that operates at the TCP layer. It is used by autonomous systems (AS), which are small units that have the authority to decide which routing protocols to use within their own systems. BGP facilitates communication between ASes. Within the realm of routing modeling, BGP introduces an important concept known as BGP community attribute which in form of '*ASN: value*'. *ASN* is AS's unique numeric mark and *value* is the community value of BGP community attribute. BGP community attributes play a crucial role in routing forwarding decisions by simplifying routing strategies and enhancing routing efficiency. They find substantial applications in optimizing network routing and monitoring network attacks, primarily due to their functions in controlling route transmission, route policies, and load balancing. The identification of BGP community attributes holds great significance in routing modeling.

However, BGP community attributes are set by network administrators, what we can obtain is number representing the AS and community value added to the corresponding routing after transmission. We cannot directly determine the specific community attributes. Moreover, even for community attributes with similar routing forwarding decisions, their corresponding value may be different. Therefore, our core research revolves around determining the specific community attributes based on the routing behaviors of routers marked with a certain identifier.

Existing methods primarily rely on rule-based designs to create dictionary-based definition for BGP community attributes. These Rule-based approaches need manual analysis to construct prior knowledge and build priori model using them to design algorithms for recognizing BGP community attributes. This results in low automation and these algorithms are applicable only to recognize specific community attributes or specific time frames. Therefore, meeting the requirements for variety and complexity in BGP community attributes becomes challenging and costly for time-sensitive applications.

In this paper, we propose a consistent and automated approach to identify BGP community attributes. We leverage graph neural networks (GNNs) to complete the recognition task by constructing a topological graph based on the routing behavior in AS-level. We first observe the structure of BGP messages forwarded by border gateway routers with different BGP communities. It turns out there are differences between message-based graph structures, for example, the Blackhole community attribute exhibits a mesh structure with traffic concentrated in a specific autonomous system, which aligns with the functionality of a Blackhole node – announcing traffic aggregation. We hypothesize that the characteristic of specific community attribute is reflected in the structured representation of the routing path. This implies that different community attributes will have significantly different graph structures. We crawl the BGP routing table from public dataset, and we acquire fields containing BGP community attributes and AS paths from BGP routing table. which is composed of hops between ASes. Based on these entries, we can construct a graph including all the ASes interacting with the AS to be recognized and all the routing behaviors among them. Since the number of ASes related to the AS

Digital Object Identifier: 10.1109/MNET.2024.3414113
Date of Current Version: 18 November 2024
Date of Publication: 17 July 2024

Yuntian Tan, Wenfeng Huang, Shen Su (corresponding author), and Hui Lu (corresponding author) are with the Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510000, China; Yang You is with NSFOCUS Inc., Beijing 100000, China.

0890-8044/24©2024IEEE

to be recognized is not fixed, we propose criteria to control the scale of graph in design. Nodes are ASes in the graph, and edges are hops between ASes, which are extracted from the AS paths in BGP routing table.

In recent years, there has been rapid development in graph neural networks. Alongside the emergence of GNNs, various advanced graph neural network models have been developed, demonstrating impressive performance in node-level or graph-level classification problems. According to our research hypothesis, we construct a directed graph using border gateway routers of autonomous systems and BGP messages. Leveraging graph neural network technology, we incorporate this abstract graph structure into the neural network model to achieve the goal of classifying autonomous system community attributes. After processing the routing nodes and BGP messages related to the target autonomous system, we obtain some structural properties of the graph, such as node in-degrees, out-degrees, edge types, etc. Additionally, we generate feature vectors as supplementary representations for classification, such as the path set in the routing table. These two components serve as inputs to the graph neural network model. After passing through residual networks and fully connected layers, the model outputs classification results, achieving the task of identifying community attributes. The experimental section demonstrates the effectiveness of our design in recognizing BGP community attributes.

Related Work

There are many studies indicate the importance of BGP community attributes [5]. Some researches are based on the security of ASes related to BGP community attributes [7], and some focus on the specific type of BGP community usages, such as IXP [6] and announcement [15]. In [14] the main discussion is about the relationship between ASes. Since BGP community attributes have important applications in network information transmission and routing decision-making, research on identifying community attributes has also achieved a certain degree of development. Previous study can be concluded as Designing Rule-based methods for BGP community.

Rule-based methods determine whether BGP community attributes exist and their specific meanings through manually written rules. This approach typically requires a deep understanding of specific attributes and identifiers of the BGP protocol. Rule-based methods generally require a deep understanding of the network and the specific situation of the BGP protocol, and to write corresponding rules, so the personnel requirements are relatively high. Moreover, due to the constant changes in the network environment and attack methods, the rules need to be constantly updated and optimized, and the maintenance cost is high. Therefore, rule-based methods are in low automation, and are more suitable for identifying and classifying BGP community attributes under specific scenarios and network environments.

For example, in the research of [1], a set of rules is designed for optimizing routing paths and selection. The technique in the paper is to infer the path of community attributes related to the entities or locations passed by the router by associating community attributes with autonomous systems, and this paper also added a set of heuristics to filter incorrect identification results introduced by improper network behavior. These heuristics will also share these autonomous systems and their BGP community attributes, and deal with inconsistent BGP error information; In [2] the method proposed by the author is only for the specific community attribute of black holes. A method that can automatically detect BGP black hole activities in the real world is developed and evaluated, and the set of rules written is applied to public and private BGP datasets. In [3], the role of community attributes is studied. Each autonomous system defines the semantics of its community attributes, and many autonomous systems blindly propagate their unknown community attributes. When community attributes are not filtered, there will be potential security vulnerabilities. This paper studies the handling of community attribute updates and filters multiple real-world BGP in experiments using predictable prefixes to mark information. It is proven that community attributes can lead to an increase in the number of information updates from the marked autonomous system and its neighboring autonomous system that neither adds nor filters community attribute information. In [4], the author proposes an algorithm that can infer the marking addition and marking clearing behavior of BGP community attributes based on autonomous systems as the basic unit. The algorithm operates in a passive manner and uses BGP to update messages.

Design

As we've opted for GNN approach for classification, it's challenging in the preprocessing of raw BGP data and the construction of the graph. Applying GNNs necessitates devising strategies to tackle this issue. In line with research on the application of GNNs in other domains, it's important to note that larger graph size don't necessarily yield better results in graph neural network models. In analyzing the dataset employed for experimentation, we've found the scale of data obtained by the collector is substantial. Therefore, there is a need to use criteria for limiting the volume of data to construct graphs.

Overall Design

We started from two steps for data processing. The first step is to use certain criteria to control the amount of data in the step of capturing data. BGP message forwarding is happening between routers all the time. Therefore, we control the scale of data from the perspective of time and space, Specifically, the control of data volume is carried out from two angles: the duration and the number of collectors capturing routing messages. The smallest unit of data we get is in days, so we selected three divisions: obtaining data according to one day, one month, and one year, with one day being the shortest length of time. For spatial constraints, RouteViews officially has a total of 52 collectors (now 46 are available). The raw data we got was in form of 52 files. This property of the data meets our requirements: On the one hand, we want to explore how scale of graph impacts the performance of GNN models;

On the other hand, we need a reference for the balance between model performance and the resource cost of data acquisition. We divided the collectors into groups according to the factors of 52, the scale of the collector groups is 1, 4, 13, 26, and 52. The greater the number of aggregation blocks, the more files are traversed, the more paths can be collected for each BGP community attribute, and the larger the size of a graph. With this strategy, we can efficiently control the size of every graph. We will present the fluctuations on the charts in the experimental section. After selecting data in the first step, the second step involves filtering out duplicate edges in graph. We use the processed data to construct input embedding. We will also share some insights and findings from the data set analysis in the experimental section.

We design the topology graph structure where each AS is a node in the graph, and the packet-hoppings between ASes are edges. Simultaneously, to identify the community attributes of multiple ASes, we chose to use GNNs for graph-level classification. We integrate all paths related to the target AS when building the graph for each target AS. The graph structure built around each AS is the basic unit of input to GNN models, and the scale of graph is controlled by the two criteria we set.

BGP Community Attributes

We categorized the community attributes of the BGP network into eight classes for experimentation and observation. These include Prepend, Announcement, Geographic, IXP, Blackhole, Local Preference-Customer, Type of Peer-Customer, Type of Peer-Provider, and we have labeled other unknown categories as 'unknown'. We will provide detailed information about the data size and its impact on model training in the experimental section.

Graph Embeddings

The data we obtained can be divided into three parts: labeled BGP community attributes, raw BGP routing table and AS topology data. We integrate and process these three components to obtain our input embeddings. We wrote Perl scripts to obtain the raw BGP routing table from collectors of RouteViews. We use the 6^{th} column and the 11^{th} column to construct input data. The 6^{th} column is AS paths (entries in form of 'ASN_0, ASN_1, ASN_2, ASN_3, …'), and the 11^{th} column is BGP community attributes (entries in form of 'ASN_0: $value_0$, ASN_0: $value_1$, ASN_1: $value_2$, …'). It should be clarified that the same ASN may pairs with different community values, and they are regarded as different community attributes. BGP community attributes is used for labeling, and AS paths are used to construct graphs. We will use two paragraphs to introduce how the input embedding is obtained.

We use labeled BGP community attributes to label the BGP community attributes we extracted. We obtain labeled BGP community attributes from research called "Revisiting the BGP communities usage". It's a paper can be found on google and matheo website. This research has a gitlab repository which provides json files for a set of BGP community attributes about specific AS. We fetched json files according to the ASN in BGP community attributes we extracted from raw data, and traversed json files for label of BGP community attributes. The label is used to generate label tensor in graph embedding. We will introduce the detailed content of the label tensor in our input embedding later.

As for graph construction, we traverse the BGP routing table by entries, collect the BGP community attributes in column 11, and create a list for each BGP community attribute. For each entry, if a BGP community attribute appear in column 11, we will add the AS path in column 6 to its list. Briefly, we aggregate AS paths according to BGP community attributes. After that, each BGP community attribute has a list of AS paths. We've mentioned that AS path is in form of 'ASN_0, ASN_1, ASN_2, ASN_3, …', which means AS path is composed of hops between ASes. In that case, we extract hops (ASN_x, ASN_y) from AS paths to an edge list with entries in form of (ASN_x, ASN_y).

Border gateway routers have different forwarding strategies and connections in real world, so the relationships between ASes also need to be distinguished to some extent. Besides, adding relationships in the graph can enhance the distinction between different community attributes and thereby improve the performance of our model. Meanwhile the relationship can extend dimension of node embedding by providing roles of ASes. We obtain AS topology data from the AS-relationship in CADIA, a public data website. The AS topology data is table with entries (ASN_x, ASN_y, relationship). The roles of ASes are providers and customers, so AS relationships can generally be divided into four categories: P2P, P2C, C2P, and unknown. We traversed AS topology data and edge list we constructed to do the matching. Now our edge list is with AS relationships, the form of entries is still (ASN_x, ASN_y, relationship).

The edge list contains duplicated edges, which increase the storage burden. But the frequency of edges is an important feature. In that case, we deduplicate and calculate frequency by entries in the edge list with AS relationships. For example, if two edges have the same source AS, target AS and relationship, then we will keep one of them, and set the frequency of this edge as 2. After that, we have the edge list with entries in form of (ASN_x, ASN_y, relationship, frequency).

We traversed processed edge list, meanwhile we generated a mapping to renumber ASN as node number starting from 0, which means every ASN corresponds to a node number. This process helps us get the ASNs as node feature. And we replace the ASNs in processed edge list with their node number, this makes our graph more normalized. We use this processed edge list ($Node_x$, $Node_y$, relationship, frequency) to construct input embeddings of nodes and edges. The process of data is visualized in Fig. 1.

In our design, we use one graph to recognize one BGP community attribute. The input graphs can be represented as a set $CAttr_i$ (i=1,2…). Each element in $CAttr_i$ represents a graph, $CAttr_i$ = {$Nodes_i$, $EdgeIndex_i$, $EdgeAttr_i$, C_Label_i}. The content of 4 embeddings in $CAttr_i$ are as follows:

- $Nodes_i$: A tensor(matrix) of ASes' features with shape [number of nodes, 5], which indicates each node has 5 features: ASN,

indegree of node, outdegree of node, whether a provider in AS relationships (1 for yes,0 for no), whether a costumer in AS relationships (1 for yes,0 for no). We traverse processed edge list ($Node_x$, $Node_y$, relationship, frequency) to obtain Nodes tensor.

- $EdgeIndex_i$: A tensor(matrix) stores all edges in graph with shape [2, number of edges], indicating that we use the two ASes of one hopping to denote a directed edge. we use the first 2 columns ($Node_x$, $Node_y$) of processed edge list to construct EdgeIndex.
- $EdgeAttr_i$: A tensor(matrix) with shape [number of edges, 2], which shows that every edge has 2 attributes: AS relationships and frequency of edges. We use the last 2 columns (relationship, frequency) of processed edge list to construct EdgeAttr.
- C_Label_i: A tensor(vector) which represents label of the community attribute to be recognized. In our design there are 9 classes of BGP community attributes, which are numbered from 0 to 8. The shape of label is [1, 9], in which the class of community attribute to be recognized will be marked as 1, whereas the others are 0.

Building the Graph Neural Network Model

We use the methods encapsulated in the PyTorch-geometric library for the design of GNN. We use residual network (ResNet) to perform hierarchical pooling on the topology graph, sum the outputs of the readout layers to obtain the whole graph's vector representation, and then input it into an MLP (Multilayer Perceptron, containing hidden layers and output layer) for graph classification task learning.

Residual networks use skip connections to design the model's hierarchical structure. Skip connections are used to add shallow layer outputs to deep layer outputs to strengthen feature expression and mitigate the impact of feature attenuation. In our design, we choose Topk-based pooling operations, i.e., each layer in the residual network ranks the importance of all nodes and discards nodes with low importance. This operation is performed by learning the importance of the nodes. The advantage of this is that it improves the fusion efficiency of distant nodes. A readout layer follows each pooling layer in this mechanism, aiming to achieve a one-time aggregation of global information for the graph at that scale. The specific implementation of the readout layer is to concatenate Global Average Pooling and Global Max Pooling, add the concatenation results of each readout layer, input it into the MLP for dimensional transformations. Readout layer and pooling later are after convolution and non-linear activation.

Our designed model is shown in Fig. 2. Firstly, we embed the structure information and node embedding of graph as input. Graph structure information is represented as G = {v, ε}. Node feature representation is $Nodes_i = \{N_1 \ldots N_n\}$, each node is represented as N for short. Secondly, the core lies in learning the structural features under specific topology structures using graph neural networks. Given the node features, we extract shallow and deep structural features of the topology through a multi-layer GNN model, i.e., using

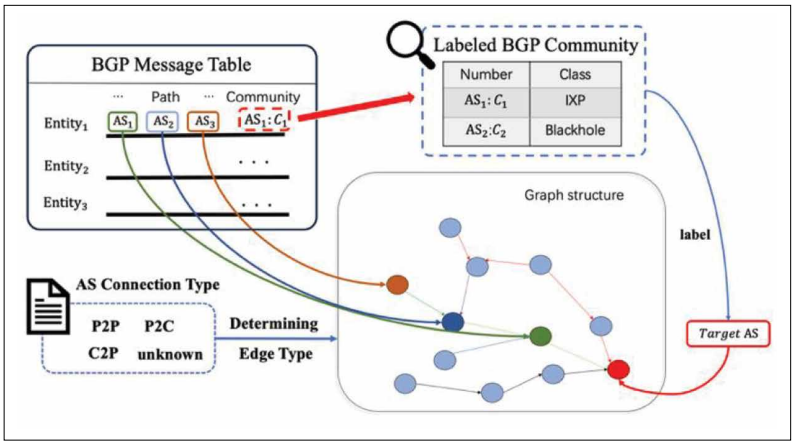

FIGURE 1. Extraction of Graph Structure from raw data.

H_V (hidden-layer of local routing node features) and H_G (hidden-layer of global routing graph features) to do forward propagation. Finally, based on the extracted graph features, the downstream decision model constructs tasks and BGP community attribute classification tasks, outputting the results. The ResNet part has five operations within three layers. The first layer performing regular convolution and non-linear activation, and the second and third layers being skip connection layers, with pooling operations additionally. The MLP has three linear transformation layers as well.

Experiment

In this section, we compare the community attribute recognition effects of different designs from the two perspectives of the data scale of graph construction and the selection of network model mentioned earlier, to obtain the most reasonable parameter setting scheme.

Dataset

In our experiments, we have chosen two public datasets as our data sources: RouteViews and RIPE. Specifically, we have written a Perl script to crawl these two datasets. When crawling the data, we input the specified time range as a parameter in the command line to obtain BGP data. The crawled data files are stored according to routes. Specifically, the RouteViews data is 199GB, the RIPE data is 45GB, and a total of 244GB of data has been collected at this stage.

According to these two division standards, we compared the data scale statistics. Specifically, we divided and counted the data scale of routing topology graphs under different time spans and different numbers of collectors, for subsequent model training, validation, and testing. We compared and discussed the model's ability to classify under different data scales. We collected data from a total of 52 collectors, analyzed the routing data for December 2017 from the public BGP data set RouteViews, and the data scale statistical results are as shown in Table 1. Table 1 is divided into two parts from top to bottom: (1) Graph statistics: number of topologies, average number of nodes, average number of edges, average number of paths; (2) Graph community classification. Table 1 is divided from left to right

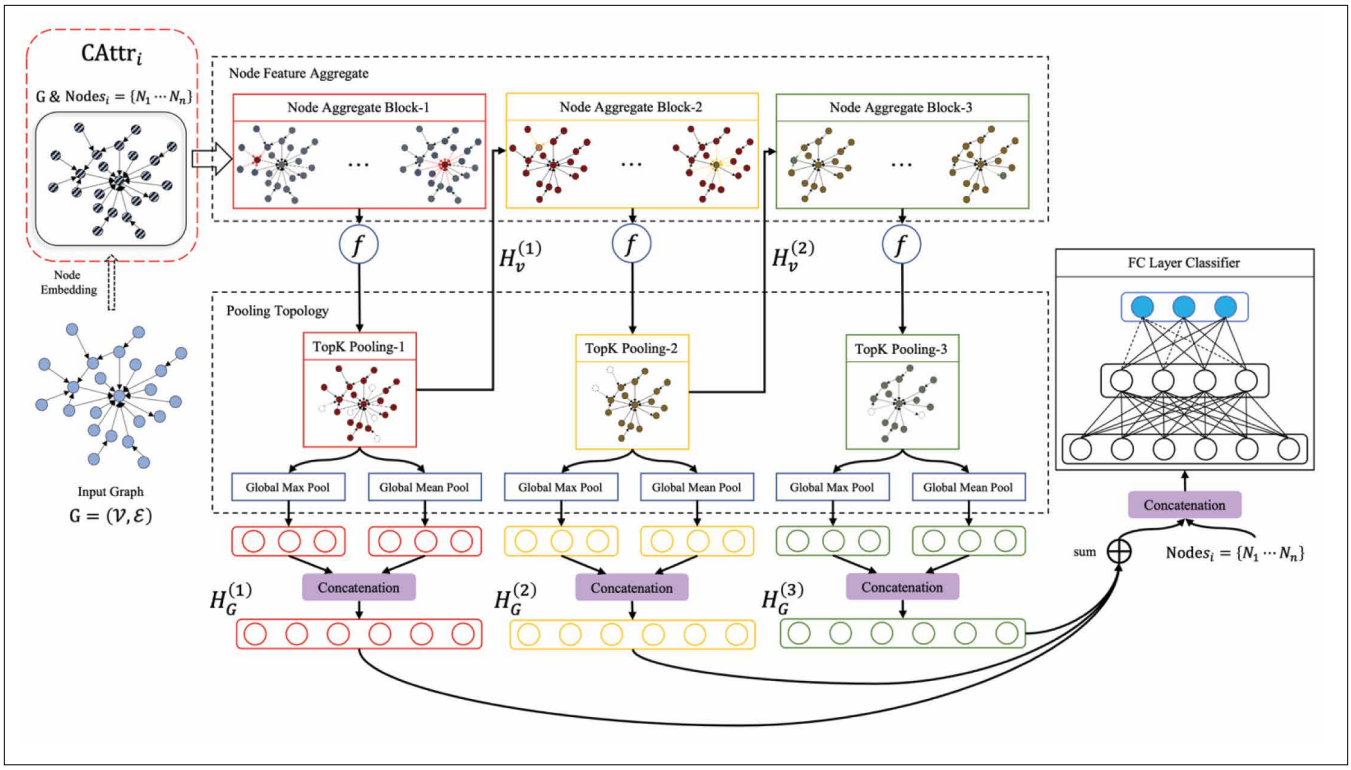

FIGURE 2. Design of neural network model.

		One Month Graph Dataset (2017.12)														
	number of Monitors	1 monitor			4 monitor			13 monitor			26 monitor			52 monitor		
	Stage	train	test	valid	train	test	valid	train	test	valid	train	test	valid	train	test	valid
Statistic of Graph	number of graphs	50530	6317	6317	68098	8513	8513	71635	8955	8955	59765	7471	7471	17015	2128	2128
	mean node number	149	153	148	110	113	110	87	88	88	77	77	78	132	132	131
	mean edge number	178	182	176	130	134	131	103	104	104	91	91	93	157	157	156
	mean path number mean	160	165	159	125	128	125	112	112	112	112	112	114	211	212	211
Classification	Prepend	8259	1008	1060	14753	1792	1850	18419	2325	2289	15047	1948	1907	3575	429	459
	Announcement	16507	2065	2105	26271	3317	3299	31705	3930	3926	30509	3741	3804	8827	1129	1084
	Geographic	20899	2632	2488	21191	2693	2624	15837	1956	2000	9612	1217	1203	2258	284	302
	IXP	2039	258	271	2115	272	256	2116	275	290	2120	247	252	1684	188	207
	Blackhole	42	7	7	78	10	7	102	13	18	102	16	16	40	8	5
	Local Preference-Customer	968	119	130	1608	181	202	1614	217	214	1160	148	127	310	41	34
	Type of Peer-Customer	1812	227	256	2067	247	272	1818	237	216	1188	152	158	313	47	36
	Type of Peer-Provider	4	1	0	15	1	3	24	2	2	27	2	4	8	2	1

TABLE 1. Data scales statistical results.

into 5 scales of collectors, and under different scales further divided into 8:1:1 for training, testing, and validation. By combining the factors of rows and columns, we find that the total amount of topology graph data aggregated by 13 collectors is the largest, and the total amount of graph data aggregated by 52 collectors is the smallest; Prepend, Announcement, and Geographic types of graphs account for the vast majority, Blackhole and Type of Peer-Provider can be almost ignored; the average number of nodes, edges, and paths under any scale are all less than 200, macroscopically most are sparse small graphs (average edge number <2 * average node number). We wrote a data screening program in Python language, and through this program, we filtered, screened, and aggregated the original BGP data, and finally generated the routing topology graph data aggregated by community attributes, which will be used as the input of the GNN models we built. When we conducted the experiments in this paper, we generated 500G of routing topology data in total.

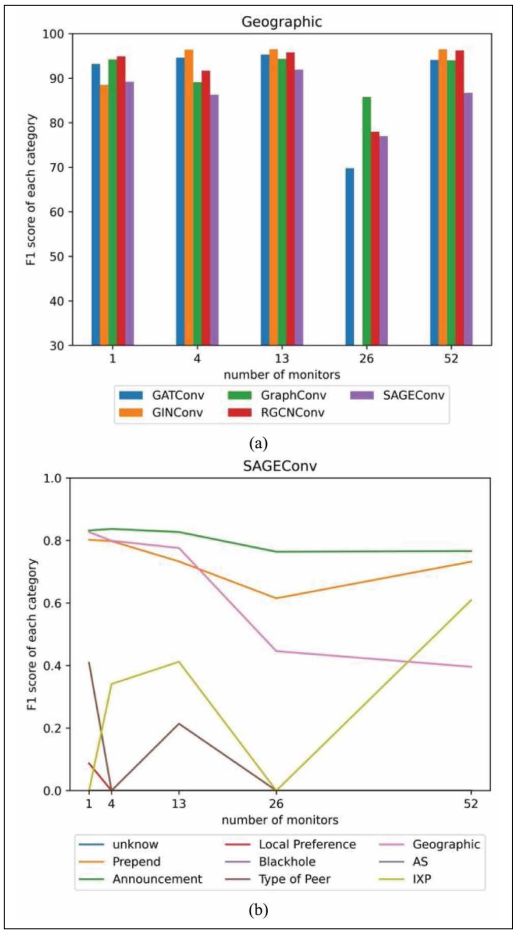

FIGURE 3. Comparison of the accuracy of some model results. a) F1 scores of different algorithms. b) F1 scores of different categories of SAGEConv.

Each graph embedding includes the 3 attributes in our experimental design (excluding the labels of community attributes).

Experimental Setup

In the experiment, as we said in the design, we first divided the data according to time length and the number of collectors. This is to construct topology graphs of different scales. The time dimension is divided into one day, one month, and one year. The number of collectors is divided into 1, 4, 13, 26, 52 per set. For the GNN model, In the MLP, we designed a convolution with non-linear activation, and applied the dropout method. This method is intended to solve the problems of model overfitting and large time overhead, the latter of which arises from a decrease in the learning rate. We set the dropout rate to 0.5 and designed it to occur after the second layer of convolution in the MLP, with the result then undergoing non-linear activation. Regarding the division of the dataset, we chose 80% as the training set, 10% as the validation set, and 10% as the test set. For the parameter settings in the model design, the optimizer we used is the Adam optimizer, the learning rate is set to 0.005, and we use a function to clear the gradient. The loss function chosen is the cross-entropy function, used to measure the training effect of the model. The ratio of TopKPooling is set as 0.8. In training, we set it to go through 20 rounds of training.

Experimental Result

The accuracy of model implementation classification and recognition is measured by the proportion of community attributes in unified model classification and F1 score. We selected four kinds of GNN algorithm models, namely: GATConv [8], GINConv [9], GraphConv [10], SAGEConv [11], for comparison with the performance of RGCNConv [12]. The main purposes of our experiment in using multiple GNN algorithms for comparison are two-fold: (1) to verify the effectiveness of modeling the identification of community attributes as a graph classification problem; (2) to deeply explore the characteristics of the routing topology graph in combination with the preferences of various algorithms in different community categories. The comparison results are as shown in Fig. 3.

Fig. 3(a) shows the differences in F1 scores of different algorithms under different scales of collectors. RGCNConv has a relatively stable prediction accuracy compared to other models and the highest accuracy when the scale of collectors is 1, 13, and 52. Although GINConv is higher than RGCNConv with 4 collectors, it even scored 0 with 13 collectors. This is a very abnormal result, to some extent, it is not as good as RGCNConv in terms of model stability. All models have F1 score of more than 90 at scale of 13 collectors, and all models perform at their worst at scale of 26 collectors. As for the zero score of GINConv, this result is abnormal and needs further research and analysis. SAGEConv is generally at a lower level in terms of scoring performance, so we have visualized the accuracy of the SAGEConv model in each category as shown in Fig. 3(b). According to the proportions of different category groups displayed in our data statistics, SAGEConv can only accurately classify categories with large amounts of data, yet the accuracy of other categories is not high. Some categories even have an F1 score of 0. It's obvious that when the scale of collectors is 13, SAGEConv has the best performance overall.

Conclusion

In this paper, we present a consistent and automated method for identifying BGP community attributes. This approach addresses the limitations of existing work, which relies on manual intervention, resulting in low automation and coarse granularity. The primary objective of our research is to identify community attributes of ASes. We capture and process BGP messages to construct a graph that characterizes the routing policies and behaviors of specific AS. Moreover, we integrate the AS relationships and labeled BGP attributes with the graph, as input embedding. Subsequently, we utilize GNN models, combined with the feature information, to learn the structure of graph, thereby accomplishing community attributes recognition. In summary, we propose an automatic interpretation method for BGP community attributes. Within this framework, we also control the graph size by collector number and time intervals. Based on this foundation, we conduct comparative experiments involving various GNN models and different

feature factors to comprehensively investigate their impact on the accuracy of community attribute recognition.

In future research, we will address several issues related to experiment and design from three main perspectives. Firstly, we will investigate the disparities in performance among different models, particularly the anomalies performance when using SAGEConv and GINConv. Secondly, we will explore the possibility of expanding the features used in our experiments. We plan to extract more route and community attribute features to improve the performance in classification. Thirdly, to expand the application scenarios of our design, there are extended concepts of BGP community attributes might be considered. The evolution of BGP communities brings new concepts like flow spec community, but presently it's lacking related work to analyze semantics of these extended communities. Related research will facilitate the performance of our design to be more applicable. Also, there are other AI methods which can be used for recognizing extended community attributes. For example, we may conduct information crawling from the ISP website and recognize the BGP community based on the LLM in our further research. The BGP ecosystem is getting more complex and crazier at present, which makes identifying complex AS relationships a particularly meaningful issue. In this paper, our method can recognize BGP community attributes, we can take the identified BGP communities with the route distribution behaviors as input features to identify complex AS relationships.

References

[1] B. A. D. Silva Jr. et al., "Automatic inference of BGP location communities," *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 50, no. 1, pp. 3–4, Jun. 2022.

[2] V. Giotsas et al., "Inferring BGP blackholing activity in the internet," in *Proc. Internet Meas. Conf.*, New York, NY, USA, Nov. 2017, pp. 1–14.

[3] T. Krenc, R. Beverly, and G. Smaragdakis, "Keep your communities clean: Exploring the routing message impact of BGP communities," in *Proc. 16th Int. Conf. Emerg. Netw. Exp. Technol.*, New York, NY, USA, Nov. 2020, pp. 443–450.

[4] T. Krenc, R. Beverly, and G. Smaragdakis, "AS-level BGP community usage classification," in *Proc. 21st ACM Internet Meas. Conf.*, Nov. 2021, pp. 577–592.

[5] V. Giotsas and S. Zhou, "Valley-free violation in internet routing—Analysis based on BGP community data," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Ottawa, ON, Canada, Jun. 2012, pp. 1193–1197.

[6] V. Giotsas and S. Zhou, "Improving the discovery of IXP peering links through passive BGP measurements," in *Proc. IEEE Conf. Comput. Commun. Workshops*, Turin, Italy, Apr. 2013, pp. 121–126.

[7] H. Birge-Lee et al., "SICO: Surgical interception attacks by manipulating BGP communities," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, New York, NY, USA, Nov. 2019, pp. 431–448.

[8] P. Velickovic et al., "Graph attention networks," *Stat*, vol. 1050, no. 20, pp. 10–48550, 2017.

[9] K. Xu et al., "How powerful are graph neural networks?" 2018, *arXiv:1810.00826*.

[10] C. Morris et al., "Weisfeiler and Leman go neural: Higher-order graph neural networks," in *Proc. AAAI Conf. Artif. Intell.*, 2019, vol. 33, no. 1, pp. 4602–4609.

[11] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[12] M. Schlichtkrull et al., "Modeling relational data with graph convolutional networks," in *Proc. 15th Int. Conf. ESWC*, Heraklion, Crete, Greece, Springer, Jun. 2018, pp. 593–607.

[13] C. Orsini et al., "BGPStream: A software framework for live and historical BGP data analysis," in *Proc. Internet Meas. Conf.*, New York, NY, USA, Nov. 2016, pp. 429–444.

[14] L. Prehn and A. Feldmann, "How biased is our validation (data) for AS relationships?" in *Proc. 21st ACM Internet Meas. Conf.*, New York, NY, USA, Nov. 2021, pp. 612–620.

[15] J. H. Park et al., "Investigating occurrence of duplicate updates in BGP announcements," in *Passive and Active Measurement* (Lecture Notes in Computer Science), vol. 6032. Berlin, Germany: Springer, 2010.

Biographies

Yuntian Tan (2112233010@e.gzhu.edu.cn) is currently pursuing the degree with the Cyberspace Institute of Advanced Technology of Guangzhou University, Guangzhou.

Wenfeng Huang (2111906044@e.gzhu.edu.cn) received the M.E. degree in computer technology from Guangzhou University, Guangzhou, China, in 2022. He is currently a Research Software Engineer with Voyager. His research interests include deep learning, autonomous driving, border gateway protocol, computer networks, and DNS.

Yang You (youyang@nsfocus.com) received the bachelor's degree in software engineering from the Beijing Institute of Technology, in 2005, and the master's degree in computer software and theory from PLA Information Engineering University, in 2008. He holds certifications as a Technology Project Manager from the Ministry of Human Resources and Social Security and has been awarded the title of Intermediate Engineer. He has held various positions in technology management and project leadership roles at institutions such as the National Information Technology Security Research Center and the Chinese Academy of Sciences. In 2017, he joined Venustech Group Company Ltd. He is currently works as the Director of the Industry Cooperation Department. He is also an off-campus supervisor for master's students with the Beijing Institute of Technology and a Senior Member of the China Computer Federation (CCF). With over a decade of experience in collaborative industry-academia-research ecosystems and research project management. He has served as the Project Leader or Key Contributor in more than ten national and local research projects. He has received the CCF Technology Award (Second Prize) once, published four high-quality papers, and participated in the application for three invention patents.

Shen Su (sushen@gzhu.edu.cn) received the B.E., M.E., and Ph.D. degrees from the Harbin Institute of Technology. He is currently an Associate Professor with Guangzhou University, Guangzhou. He has published more than 60 journal articles and conference papers in such areas, with more than 2100 citations and seven ESI high-indexed papers. His research interests include blockchain security, DNS, route modelling, route security, cyber range, vehicular networks, and wireless sensor networks. He has served as a Guest editor of IEEE Network and CMES, and a reviewers for IEEE Transaction on Industrial Information, IEEE *Network Magazine*, and IEEE Internet Computing, Journal of the Franklin Institute. He has also served as the Owner and a Key Staff in a number of projects from the National Natural Science Foundation of China, the National Key Research and Development Program, and the National 973 Project.

Hui Lu (luhui@gzhu.edu.cn) received the Ph.D. degree in engineering from the Beijing University of Posts and Telecommunications. He is currently a Professor and the Director of Fang class with the Guangzhou University, Cyberspace Institute of Advanced Technology, the Secretary-General with the Cyberspace Security Talent Education Alliance of China(CEAC), the Director of the Competition and Exercise Working Committee of Cybersecurity Association of China (CSAC), the Executive Member of the 11th competition committee of the Guangdong Computer Academy, a talent listed into "100 Talents Program" of Guangzhou University, an Expert Member of Ping An Academy of Financial Security, and a member of the technical committee of the big data security talent cultivation base of Cybersecurity Association of China. He is currently chairing a general program under the National Natural Science Foundation of China (NSFC), and responsible for the Guangdong Provincial Key Area Research and Development Program "Key Technologies for Automated Cyber Attack and Defense of New Infrastructure" as the executive director. As a representative of his research team, he established the "Guangdong Province Joint Postgraduate Training Demonstration Base" with the Peng Cheng Laboratory. His research results have been published in more than 60 papers in domestic and international journals and conferences. His research interests include artificial intelligence security, network security, and vulnerability mining.