# BEAR: BGP Event Analysis and Reporting

1st Hanqing Li
*Engineering Sciences and Applied Mathematics Department*
*Northwestern University*
Evanston, USA
hanqingli2025@u.northwestern.edu

2nd Melania Fedeli
*Amazon Web Services (AWS)*
Dublin, Ireland
melaniaf@amazon.com

3rd Vinay Kolar
*Amazon Web Services (AWS)*
Cupertino, USA
vinkolar@amazon.com

4th Diego Klabjan
*Industrial Engineering and Management Sciences Department*
*Northwestern University*
Evanston, USA
d-klabjan@northwestern.edu

***Abstract*—The Internet consists of interconnected, independently managed Autonomous Systems (AS) that rely on the Border Gateway Protocol (BGP) for inter-domain routing. BGP anomalies—such as route leaks and hijacks—can divert traffic through unauthorized or inefficient paths, jeopardizing network reliability and security. Although existing rule-based and machine learning methods can detect these anomalies using structured metrics, they still require experts with in-depth BGP knowledge of, for example, AS relationships and historical incidents, to interpret events and propose remediation. In this paper, we introduce BEAR (BGP Event Analysis and Reporting), a novel framework that leverages large language models (LLMs) to automatically generate comprehensive reports explaining detected BGP anomaly events. BEAR employs a multi-step reasoning process that translates tabular BGP data into detailed textual narratives, enhancing interpretability and analytical precision. To address the limited availability of publicly documented BGP anomalies, we also present a synthetic data generation framework powered by LLMs. Evaluations on both real and synthetic datasets demonstrate that BEAR achieves 100% accuracy, outperforming Chain-of-Thought and in-context learning baselines. This work pioneers an automated approach for explaining BGP anomaly events, offering valuable operational insights for network management.**

***Index Terms*—large language models (LLM), border gateway protocol (BGP), anomaly explanation, prompt engineering**

## I. Introduction

The Border Gateway Protocol (BGP) is the de facto inter-domain routing protocol that enables Autonomous Systems (ASes) to exchange reachability information and maintain global Internet connectivity [1], [2]. However, BGP was not designed with built-in route authentication, making it vulnerable to anomalies such as prefix hijacks and route leaks [3]. In a hijack, an AS illegitimately announces reachability to a prefix it does not own; in a route leak, an AS propagates routes in violation of routing policies. Such events can disrupt connectivity and require timely diagnosis and mitigation.

Existing work has extensively studied BGP anomaly detection using rule-based, statistical, machine learning, and graph-based approaches [3]–[7]. These methods primarily answer whether an anomaly occurred. In practice, however, operators also need to understand what happened: the event type, the affected prefix and victim AS, the malicious or misconfigured AS, and the pre- and post-event AS-path changes. We refer to this downstream task as *BGP anomaly event explanation* and define it formally in Section III-A.

Large Language Models (LLMs) have demonstrated strong capabilities in automatic text generation, instruction following, and complex reasoning [8], making them promising for explaining BGP anomaly events. Existing rule-based and machine learning methods are effective for detecting anomalies from predefined BGP features [3], but they typically do not generate operator-facing explanations that connect AS-path changes with event types, relevant ASes, and routing-domain knowledge. At the same time, directly applying LLMs to raw BGP data is challenging because each event may involve thousands of structured AS paths across prefixes, peers, and collectors, and LLMs are not primarily designed for precise reasoning over large-scale structured numerical data [9]. Thus, rather than replacing existing detectors, BEAR complements them by employing LLMs to interpret and explain detected anomalies in detail.

In this paper, we propose BEAR (BGP Event Analysis and Reporting), an LLM-based framework for generating comprehensive reports that explain detected BGP anomaly events. Given an event timestamp and affected IP prefix, BEAR retrieves relevant BGP data from public routing repositories and constructs historical, pre-event, and post-event AS-path views. To align BGP data with the strengths of LLMs, BEAR adopts a multi-step reasoning approach that first transforms large-scale tabular routing data into textual path-change summaries and then uses the LLM to infer the event type, identify relevant ASes, and generate the final report. We further improve report accuracy through prompt augmentation and self-consistency, and evaluate BEAR under partial collector availability to study its robustness in realistic monitoring settings. We focus on two primary types of BGP anomalies, as categorized in Al-Musawi et al.'s taxonomy [3]: (1) direct intended anomalies (e.g., BGP hijacks) and (2) direct unintended anomalies (e.g., BGP route leaks). These two categories are prioritized because they are operationally important and can be analyzed using publicly

available routing data.

To address the scarcity of fully documented BGP anomaly events, we also introduce an LLM-assisted framework for generating synthetic BGP anomaly data. The framework modifies realistic BGP routing states according to LLM-generated anomaly specifications, yielding controlled events with known ground truth. We evaluate BEAR on real, anonymized real, and synthetic BGP anomaly events, with generated reports assessed against expert-verified labels. BEAR achieves 100% accuracy on both real and synthetic datasets.

Our contributions are listed next.

- **Problem Definition and Exploration:** This work is the first to introduce and investigate the problem of BGP anomaly event explanation, aiming to provide insights from both structured data and LLM global knowledge into events by analyzing BGP data both before and after the anomaly event.
- **Novel Multi-Step Reasoning Approach:** The paper proposes a novel multi-step reasoning approach by transforming large-scale tabular BGP data into textual descriptions for enhanced interpretability and performing data analysis and reasoning on the text description instead of the tabular data for high reasoning accuracy. By leveraging prompt augmentation and self-consistency mechanisms, the approach achieves 100% accuracy and generates detailed BGP anomaly event reports, effectively identifying missing key elements such as event types, relevant ASes, and comprehensive explanations.
- **Robustness of Availability of Collectors**: In practice, collectors may be unavailable. We study such an impact by creating settings with limited availabilities of collectors. With respect to LLMs, this accommodates the use of LLMs with a limited number of input tokens. Leveraging fewer collectors also results in a faster explanation of the events and a quicker mitigation and remediation, thus reducing network impacts.
- **Synthetic Data Generation Framework:** In this research, an innovative framework is presented that utilizes LLMs to synthesize high-quality synthetic BGP anomaly event datasets. This framework generates critical features such as event types, relevant ASes, target IPs, timestamps, and AS paths before and after the events, marking the first method designed to synthesize BGP anomaly events.

All code is available at https://github.com/hanklee97121/BEAR_BGP_EVENT_ANALYSIS.

## II. Related Research

This section reviews related literature on BGP anomaly detection and the applications of LLMs in the BGP domain.

### A. BGP Anomaly Detection

Prior work has extensively studied BGP anomaly detection using rule-based, statistical, machine learning, time-series, and graph-based methods [3]–[7]. These methods detect routing disruptions such as prefix hijacks, route leaks, and abnormal update patterns from BGP features, AS-path changes, reachability signals, or graph representations. BEAR is complementary: rather than detecting whether an anomaly occurred, it explains a detected event by identifying the event type, relevant ASes, and supporting AS-path evidence.

### B. LLMs for Networking

LLMs have recently been explored for networking tasks such as router configuration synthesis, network analysis, and command-line assistance [10]–[12]. These works demonstrate the potential of LLMs to assist network operators, but they do not address the specific challenge of analyzing BGP anomaly events and generating comprehensive explanatory reports. Our approach bridges this gap, introducing the first method explicitly designed for BGP anomaly event explanation.

## III. Methodology

This section outlines the methodology for generating the BGP anomaly event report. First, we formally define the BGP anomaly event explanation problem. Next, we describe the process of retrieving the relevant data for an event analysis. Finally, we present the framework for producing high-quality BGP anomaly event reports.

### A. Problem Definition

Before introducing the method, we formally define the problem of BGP anomaly event explanation. A BGP anomaly event $E$ is given by an associated IP prefix $ip$ and a specific start time $t$. We assume this event is currently detected by any of the BGP anomaly detection algorithms, such as the ones explained in Section II-A. Additionally, we have access to the BGP dataset $D_{BGP}$, which contains BGP update messages and routing information over time. The objective is to develop a function $f$ that processes this information to generate a natural language report $R$. The report should explain the details of the event $E$ and identify key unknown features, such as the event type and relevant ASes.

### B. BGP Data Retrieval

Given the affected prefix $ip$ and event start time $t$, BEAR uses BGPStream [13], an open-source framework for accessing historical and live BGP data from public collectors, to retrieve routing information relevant to the event. A collector receives BGP data from multiple peers, and each peer provides an AS path, i.e., a sequence of ASes used to reach a prefix. BEAR uses two types of BGP records: RIB records, which are periodic snapshots of active routing tables, and BGP update messages, which report routing changes, such as path announcements and withdrawals.

**Historical view:** $D_{history}$ contains AS-paths from RIB records to $ip$ from at least eight hours before the event. This view provides a stable reference that is not affected by the anomaly.

**Pre-event view:** $D_{before}$ is constructed by updating $D_{history}$ with BGP update messages observed from the timestamp of $D_{history}$ up to five minutes before the event start time,

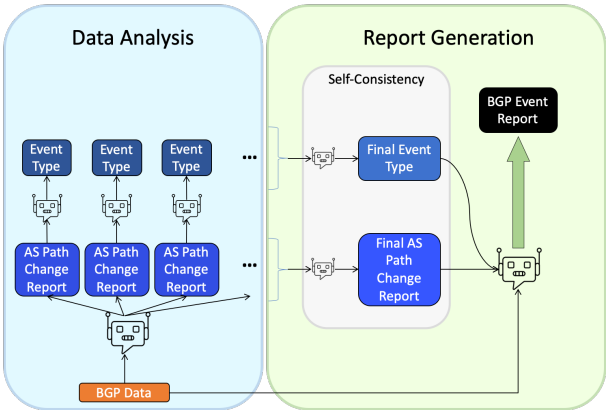


Fig. 1. Overview of the framework that generates report for BGP anomaly events

i.e., $t-5$ minutes. This produces the routing state immediately before the anomaly.

**Post-event view:** $D_{after}$ is constructed by further updating $D_{before}$ with BGP update messages observed from $t-5$ minutes to $t+5$ minutes, or to one second before the event end time if the event ends earlier. This produces the routing state after the anomaly begins.

To capture sub-prefix hijacks and route leaks, BEAR also retrieves more-specific and less-specific prefixes related to $ip$. The resulting data are represented as collector-peer-AS-path mappings and used as input to the analysis stage.

### C. Data Analysis and Report Generation

Most IP prefixes are associated with between 1,000 and 1,200 recorded AS paths, and each event involves, on average, 2 to 3 IP prefixes in the BGP data. Given the scale of the dataset—which comprises up to 1,545 peers, with most events recording between 2,000 and 3,600 AS paths—directly analyzing these three datasets to produce an accurate explanation presents a significant challenge for LLMs. To address this, we employ a multi-step reasoning framework that systematically guides the LLM through the report generation process, incorporating prompt engineering, in-context learning, and a self-consistency mechanism to enhance accuracy and reliability.

The method begins with a data analysis phase designed to transform tabular BGP data into textual descriptions, as illustrated in Figure 1. The LLM is first provided with the BGP data in a tabular format and prompted to describe changes in AS paths by comparing $D_{after}$ with $D_{before}$, using $D_{history}$ as a reference. The prompt is carefully structured to decompose the reasoning process, directing the LLM to analyze AS path changes systematically. The LLM is required to answer the following key questions.

- Does the existing path from each peer to the target IP prefix change?
- If it does, does the last AS (destination) change or not?
- Is there any new AS path to a new sub-prefix introduced?
- If there is, compare it to the existing path with the same peer, is there any difference?
- Does the last AS (destination) change or not?

These targeted questions guide the LLM to extract meaningful patterns from the BGP data and produce structured responses. In this manner, the tabular BGP data is transformed into descriptive text containing meaningful information, which is then utilized by the LLM to determine the event type and compose the report.

Despite these refinements, we observe that the LLM occasionally misidentifies the destination AS in an AS path. For example, in the AS path [4608, 1221, 4637, 15169], the LLM may incorrectly infer AS4637 as the destination instead of the correct answer, AS15169. To mitigate this, we employ few-shot in-context learning, providing the LLM with examples of AS paths alongside their correct destination AS, ensuring it learns to accurately interpret AS paths.

Next, the LLM is tasked with classifying the anomaly type—determining whether the event is a BGP hijack or a BGP route leak based on observed AS path changes. The prompt explicitly describes the distinguishing characteristics of these anomalies, ensuring the model applies the correct classification criteria. Additionally, we emphasize in the prompt that the consequences of these anomalies may be reflected in a single AS path or a sub-prefix, ensuring that the model accounts for sub-prefix anomalies in its analysis. This emphasis also reinforces the importance of detecting even subtle differences between $D_{before}$ and $D_{after}$, enabling the LLM to identify minor but crucial changes that may indicate an anomaly.

To improve robustness, we repeat this data analysis and classification processes identically $N$ times, generating $N$ AS path change reports and $N$ anomaly classifications. They differ due to the stochastic generation process of LLMs. A self-consistency mechanism is then applied, where we ask the LLM to select the most frequently occurring anomaly classification across all iterations. Then, instead of directly choosing one of the generated AS path change reports, we prompt the LLM to synthesize the final AS path change report that aligns with the majority of the previously generated reports, ensuring consistency and coherence. This approach mitigates variability in the LLM's outputs and reinforces correctness by prioritizing the most consistent and frequently observed description.

Finally, we provide the BGP data, the final AS path change report, and the final anomaly classification as inputs to the LLM, prompting it to generate a comprehensive BGP event report. The crafted prompt ensures that the report includes key details such as the event type and affected ASes. By combining multi-step reasoning, prompt engineering, and self-consistency, BEAR produces highly accurate and interpretable reports that effectively explain BGP anomaly events.

## IV. Dataset

We gather information on 10 well-documented BGP anomaly events from online sources, recording their event type, relevant ASes, target IP, start time, end time, event name, and a link to the event description. The corresponding BGP data for each event is retrieved as outlined in Section III-B.

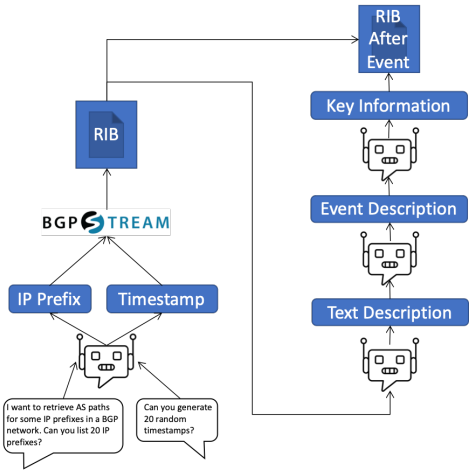


Fig. 2. Workflow of synthetic BGP event generation.

Given the detailed online descriptions of these events, many of which occurred prior to 2023, it is possible that the LLM used in our experiments has been trained on these reports. To ensure that the LLM generates explanations by reasoning and analyzing data rather than retrieving reports from its memory, we modified the BGP data for these 10 events. Specifically, we replaced the AS numbers with random values and the timestamps with random timestamps, creating 10 additional samples. These modified events retain the same structural characteristics as the original events but eliminate identifying features that could trigger memory-based retrieval by the LLM. The resulting dataset consists of 20 samples, encompassing both the original events and their anonymized counterparts, which we refer to collectively as the "real events."

### A. Synthetic BGP Event

Due to the scarcity of publicly documented BGP anomaly events, we construct a synthetic dataset with controlled ground truth. As shown in Figure 2, the goal is to preserve realistic pre-event routing states while generating post-event routing changes that simulate BGP anomalies.

For each synthetic event, we first prompt the LLM to generate a valid IP prefix and timestamp, which are used to retrieve routing information via BGPStream and construct $D_{history}$ and $D_{before}$. To ensure that the retrieved data are not affected by an anomaly, we verify the consistency of AS paths between these two views. We then provide $D_{before}$ to the LLM and ask it to summarize AS-path patterns and representative examples. Based on these patterns, the LLM generates a hypothetical anomaly event description, including the event type, applicable sub-prefix, hijacker or leaker, representative affected AS paths, and the percentage of peers detecting the event.

Using the generated description, we extract the key information and modify $D_{before}$ to create $D_{after}$. For sub-prefix hijacks or route leaks, paths for the target prefix are first duplicated to the sub-prefix, and subsequent edits are applied to the sub-prefix data. Peers are sampled according to the generated detection percentage, and their AS paths are replaced with LLM-generated examples. For hijacks, the modified path starts with the peer and ends with the hijacker; for route leaks, the path suffix is replaced with a path starting from the leaker and ending at the legitimate destination AS. After these modifications, $D_{after}$ is finalized as the post-event routing state.

We generate 34 synthetic BGP anomaly events, evenly divided into 17 BGP hijacks and 17 BGP route leaks, to assess our method. This dataset is referred to as the "synthetic" dataset.

TABLE I
NUMBER OF CORRECT REPORTS OUT OF 54 EVENTS (10 REAL + 10 ANONYMIZED REAL + 34 SYNTHETIC). EACH CELL SHOWS *Total* AND THE BREAKDOWN *(Real + Anonymized + Synthetic)*.

| Method | Event type | Malicious AS | Victim AS |
|---|---|---|---|
| CoT | 41 (5+9+27) | 47 (8+6+33) | 45 (7+6+32) |
| In-Context | 43 (8+9+26) | 49 (8+8+33) | 46 (7+7+32) |
| CoT+In-Context | 44 (8+9+27) | 49 (7+10+32) | 46 (7+7+32) |
| $BEAR_0$ | 51 (9+9+33) | 49 (7+9+33) | 50 (9+8+33) |
| BEAR | 54 (10+10+34) | 54 (10+10+34) | 54 (10+10+34) |

```
--- ## BGP Anomaly Event Report ### Event Overview - **Event Type:** BGP Hijack - **Target IP Prefix:** 205.251.192.0/23 - **Timestamp:** 2018-04-24 11:05:00 ### Detail Analysis #### 1. Previous And Recent AS Path Observations - **Historical AS Paths for 205.251.192.0/23:** - The destination AS consistently recorded was `16509` across various collectors, indicating stable routing towards the legitimate network owner. - The paths to the main prefix `205.251.192.0/23` were well established through major AS paths such as `3356`, `2914`, and `1299` to `16509`. - **Observed AS Path Changes Post-Anomaly:** - There were no changes in the routing paths for the primary prefix `205.251.192.0/23`, and the paths continued ending with AS `16509`. - **Introduction of New AS Paths for Sub-prefixes:** - The sub-prefixes `205.251.192.0/24` and `205.251.193.0/24` appeared with new AS paths ending at AS `10297`. - Collectors such as RRC00, RRC03, RRC06, RRC07, RRC11, RRC14, RRC16, RRC19, RRC20, and RRC23 identified these sub-prefixes' announcements with significant route changes. - The presence of AS `6939` as a common path element before reaching AS `10297` was noted across multiple collectors, suggesting a potential intervention route towards these IPs. #### 2. Anomaly Characterization - **Change Implications:** - The key indicator of a BGP hijack arises from the new routes diverting traffic meant for the legitimate AS `16509` to AS `10297` for sub-prefixes. - Despite the main prefix's unchanged routing, the emergence and redistribution of traffic paths for `205.251.192.0/24` and `205.251.193.0/24` highlight potential unauthorized route announcement or traffic diversion. #### 3. Technical and Tactical Considerations - **Technical Explanation:** - A BGP hijack occurs when incorrect announcements manipulate routers globally to reroute traffic away from its intended paths, often leading traffic to an unauthorized network or entity. - Malicious intent or misconfiguration can cause such route deviations, leading to data interception, traffic analysis, or denial of service due to improper routing paths. - **Involved Autonomous Systems (AS):** - Original Destination: AS `16509` - Suspected Detour Destination: AS `10297` - Frequent AS in Routing Path: AS `6939` ### Conclusion and Recommendations - **Conclusion:** - No significant deviation was observed in paths ending with the primary IP prefix. Still, the emergence of new paths for its sub-prefixes with different destination AS indicates a likely BGP hijack event, redirecting traffic away from AS `16509` to AS `10297`. - **Recommendations:** - Immediate coordination with AS `10297` and involved upstreams like AS `6939` for resolution. - Implement RPKI (Resource Public Key Infrastructure) and monitoring systems to mitigate similar incidents. - A detailed review of BGP announcements and validation of routing policies to prevent future occurrences. --- This report elaborates on the observed BGP anomaly event, providing a comprehensive analysis and potential mitigation stance based on the given timing and AS path changes detected.
```

Fig. 3. An example report for a BGP hijack event. In this report, BEAR not only identifies the event type but also detects the hijacked sub-prefix.

## V. EXPERIMENT RESULTS

We evaluate BEAR on 54 events, including 10 real, 10 anonymized real, and 34 synthetic events. We set $N = 5$ for self-consistency and use GPT-4o as the backbone LLM. We compare BEAR with chain-of-thought (CoT) prompting [14], in-context learning [15], their combination (CoT+In-Context), and BEAR without self-consistency ($BEAR_0$). CoT provides definitions of BGP hijacks and route leaks and asks the LLM to reason step by step. In-Context provides four synthetic examples covering hijack, sub-prefix hijack, route leak, and sub-prefix route leak; these examples are excluded from the synthetic test set. We evaluate whether each report correctly identifies the event type, malicious AS, and victim AS; a report is fully correct only if all three elements are correct.

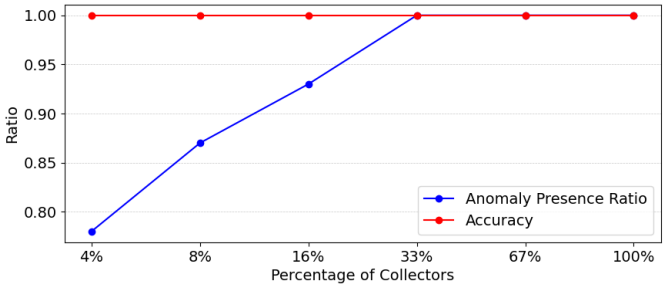


Fig. 4. Experimental results under limited data conditions (x-axis shows the percentage of collectors). The anomaly presence ratio represents the percentage of anomaly events captured in $D'_{before}$ and $D'_{after}$. Accuracy counts reports that either correctly identify all three key elements of an anomaly event or, when event-related data are missing, produce an inconclusive report that recommends collecting additional data from other collectors.

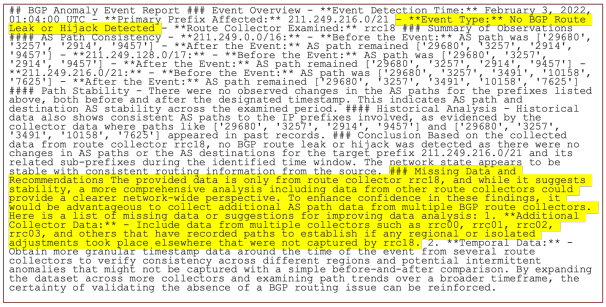


Fig. 5. Example report when $D'_{before}$ and $D'_{after}$ does not contain the anomaly event.

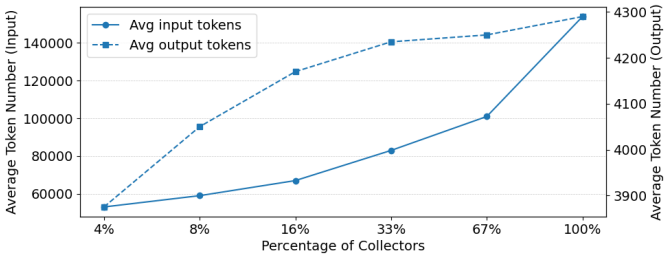


Fig. 6. Average number of tokens required to generate a report at different levels of available data. The x-axis represents the percentage of collectors. The two lines show how resource requirements vary as the amount of provided data increases: one line displays the number of input tokens, and the other line shows the number of output tokens.

As shown in Table I, BEAR correctly identifies all three elements for all 54 events, outperforming all baselines. $\text{BEAR}_0$ also substantially improves over CoT, In-Context, and CoT+In-Context, showing the substantial gains in data analysis enabled by the multi-step reasoning approach. The improvement from $\text{BEAR}_0$ to BEAR further demonstrates the value of self-consistency. Figure 3 shows an example BEAR report, illustrating that the generated output includes the event type, malicious AS, victim AS, affected sub-prefix, other relevant ASes, supporting evidence, and recommended mitigation actions.

While historical data offers near-complete visibility, real-time BGP visibility can be incomplete because public collection platforms observe only a subset of AS links and collectors may be temporarily unavailable. Alfroy et al. [16] show that only about 16% of peer-to-peer links are observed in real time, causing roughly 24% of simulated BGP hijacks to be missed. Motivated by this limitation, we evaluate BEAR under partial collector availability. We randomly select a subset of the 24 RIPE RIS collectors and restrict $D_{before}$ and $D_{after}$ to the selected collectors, yielding $D'_{before}$ and $D'_{after}$, while keeping $D_{history}$ as a full historical reference.

The desired behavior is twofold: when anomaly evidence is present in $D'_{before}$ and $D'_{after}$, BEAR should generate a correct report; when the evidence is absent, it should return an inconclusive report and recommend collecting additional data. As shown in Figure 4, the anomaly presence ratio increases with collector availability, while BEAR maintains 100% correctness under this criterion. With one collector, 78% of events are captured; with two and four collectors, the capture rates increase to 87% and 93%, respectively. When 33% or more collectors are available, all events are captured. For events not present in the limited views, BEAR produces inconclusive reports and recommends additional data collection, as illustrated in Figure 5.

Additionally, we evaluate the token resources required for report generation under different levels of collectors, as illustrated in Figure 6. The token counts represent the average over all 54 events (both real and synthetic), as resource usage is similar across real and synthetic events. In general, the total token count used by BEAR to generate a BGP anomaly explanation increases sublinearly with the number of collectors—and, consequently, with the data volume. When employing all 24 collectors, BEAR requires an average of 153,628 input tokens and 4,285 output tokens per report. In contrast, when only 4% of collectors are available (i.e., one collector), BEAR uses an average of 53,482 input tokens and 3,874 output tokens per report.

Moreover, we assess BEAR using different LLMs, with the results depicted in Table II. When employing Claude-3.5-Sonnet, BEAR attains an accuracy of 98%, with only one anonymized real event misreported, demonstrating performance comparable to BEAR utilizing GPT-4o. In contrast, BEAR achieves only 80% accuracy with Llama-3.3-70B-Instruct, which is lower than the performance observed with GPT-4o but still surpasses the baseline methods implemented with GPT-4o. With Claude-3.5-Sonnet, BEAR processes an average of 151,923 input tokens and 3,065 output tokens per BGP report. In the case of Llama-3.3-70B-Instruct, BEAR processes an average of 110,272 input tokens and 4,338 output tokens per report. Since BEAR depends on the reasoning capabilities of LLMs, those that excel in this area are more appropriate for explaining BGP anomalies. As demonstrated in Table II, employing an LLM with reasoning abilities at least comparable to GPT-4o is essential for ensuring the accuracy of BGP report generation. Moreover, in practical deployments, a trade-off between accuracy and cost should be considered.

TABLE II
CORRECT REPORTS BY BACKBONE LLM OVER 54 EVENTS (10 REAL + 10 ANONYMIZED REAL + 34 SYNTHETIC). EACH CELL SHOWS *Total* AND *(Real + Anonymized + Synthetic)*.

| LLM | # Correct reports |
|---|---|
| GPT-4o | 54 (10+10+34) |
| Claude-3.5-Sonnet | 53 (10+9+34) |
| Llama-3.3-70B-Instruct | 43 (8+7+28) |

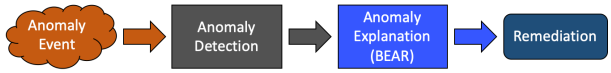


Fig. 7. A diagram to the network operations pipeline.

## VI. CONCLUSION AND FUTURE DIRECTIONS

This paper presents BEAR, a novel framework that addresses the critical challenge of explaining BGP anomaly events. By leveraging the capabilities of LLMs, BEAR transforms raw BGP data into detailed, accurate reports that provide valuable insights for network operators. Our framework's multi-step reasoning approach and integration of self-consistency mechanisms enable high accuracy in anomaly event explanation, even when data availability is limited. Furthermore, this paper presents a synthetic data generation framework that expands evaluation possibilities, mitigating the scarcity of real-world datasets. Experiments demonstrate BEAR's superior performance compared to naive methods, achieving 100% accuracy on both real and synthetic datasets. This work not only advances the state of BGP anomaly detection and explanation but also lays the foundation for integrating LLMs into broader network management applications.

Looking ahead, a promising future direction is the development of a comprehensive network operations pipeline, as in Figure 7, that integrates anomaly detection, BGP explanation, and remediation. In this envisioned pipeline, detected BGP anomalies would be automatically fed into the BEAR framework, which would generate interpretations and recommended mitigations. These outputs would then be vetted by network operators and implemented to minimize disruption.

Another possible future direction would be "agentifying" the solution so having an AI agent with multiple tools or even incorporating BEAR into a multi-agent collaboration solution for network monitoring and remediation.

Additionally, BEAR's approach could be generalized beyond networking, particularly by extending graph-based reasoning to other domains, such as social network analysis and financial fraud detection.

Finally, future work may focus on optimizing BEAR's cost-efficiency, refining its prompts to further reduce inference costs, and expanding its applicability to other network protocols and security applications.

## REFERENCES


[1] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (BGP-4)," *No. rfc4271*, 2006.

[2] C. Labovitz, G. R. Malan, and F. Jahanian, "Internet routing instability," *IEEE/ACM Transactions on Networking*, vol. 6, no. 5, pp. 515–528, 1998.

[3] 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.

[4] B. A. Scott, M. N. Johnstone, P. Szewczyk, and S. Richardson, "Matrix profile data mining for BGP anomaly detection," *Computer Networks*, vol. 242, p. 110257, 2024.

[5] H. Latif, J. Paillissé, J. Yang, A. Cabellos-Aparicio, and P. Barlet-Ros, "Unveiling the potential of graph neural networks for BGP anomaly detection," *Proceedings of the 1st International Workshop on Graph Neural Networking*, pp. 7–12, 2022.

[6] S. Peng, J. Nie, X. Shu, Z. Ruan, L. Wang, Y. Sheng, and Q. Xuan, "A multi-view framework for BGP anomaly detection via graph attention network," *Computer Networks*, vol. 214, 2022.

[7] J. Huang, M. Odiathevar, A. Valera, J. Sahni, M. Frean, and W. K. Seah, "Realtime BGP anomaly detection using graph centrality features," *International Conference on Advanced Information Networking and Applications*, pp. 222–233, 2024.

[8] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong *et al.*, "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[9] Z. Zhou and R. Yu, "Can LLMs understand time series anomalies?" *The Thirteenth International Conference on Learning Representations*, 2025.

[10] R. Mondal, A. Tang, R. Beckett, T. Millstein, and G. Varghese, "What do LLMs need to synthesize correct router configurations?" *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks*, pp. 189–195, 2023.

[11] K. B. Kan, H. Mun, G. Cao, and Y. Lee, "Mobile-llama: Instruction fine-tuning open-source llm for network analysis in 5g networks," *IEEE Network*, 2024.

[12] M. Palmero, K. P. Annamalai, H. Singaravelan, D. Zacks, and J. W. Capobianco, "Providing an ai-enabled network assistant for command line interface environments," *Technical Disclosure Commons*, 2024.

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

[14] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[15] Q. Dong, L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, B. Chang *et al.*, "A survey on in-context learning," *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 1107–1128, 2024.

[16] T. Alfroy, T. Holterbach, T. Krenc, K. Claffy, and C. Pelsser, "The next generation of BGP data collection platforms," *Proceedings of the ACM SIGCOMM 2024 Conference*, pp. 794–812, 2024.