CSC 7210: Project 2 Report

Ekle, Ocheme Anthony

November 27, 2023

Abstract– In this project, I replicated the SpotLight algorithm by Eswaran et al. [1] for detecting anomalies in streaming graphs.

1 Introduction

Anomaly detection spans various applications, from identifying intriguing patterns in relevant domains to detecting fraud and uncovering insider threats. It is a crucial task in both static and dynamic graphs. Unlike static graphs, where the topology remains constant, dynamic networks are in constant flux, changing their nodes and edges. In this project, we aim to replicate and analyze the SpotLight[1] paper on anomaly detection in streaming graphs.

1.1 Summary of Anomaly Detection Paper (SpotLight)

The paper is titled *"SpotLight: Detecting Anomalies in Streaming Graphs"* proposed by Eswaran et al. [1], a randomized sketching-based method for detecting sudden changes in dynamic graphs and detecting the appearance and disappearance of dense subgraphs or bi-cliques using sketching. The problem SpotLight set out to solve is stated below:

Problem 1. Given a stream of weighted, directed/bipartite graphs, $\{G1, G2, ...\}$, ***detect in near real-time*** whether G_t contains a sudden (dis)appearance of a large dense directed subgraph using sub-linear memory.

SpotLight assumes that an anomalous graph is mapped **'far'** away from **'normal'** instances in the sketch space with a high probability for an appropriate choice of parameters. This is done by creating a K-dimensional sketch that comprises K-subgraphs, thereby enabling the detection of sudden changes within the dynamic graph.

The code is implemented in Jupyter Notebook and takes approximately 1 hour and 30 minutes to compile.

2 Method and Paper Implementation

2.1 Data colection and Visualization

Darpa dataset contains 4.5 million IP-IP communications, involving 9484 source IPs and 23398 destination IPs across 87.7K time steps (minutes). Each communication is represented as a directed edge (srcIP, dstIP, 1, time).

I obtained seven weeks of training data and produced 730 graphs by aggregating edges in hourly durations. The dataset comprises 89 known network attacks, including portsweep, ipsweep, mscan, and snmpgetattack. Most attacks, although large ($>$ 100 edges), are targeted by a few hosts and occur in single or multiple bursts, leading to the sudden (dis)appearance of large, dense subgraphs. We label a graph as anomalous if it contains at least 50 attack edges, based on the provided ground truth.

2.2 SpotLight Algorithm and Method

The SpotLight Algorithm works in two main steps. First, it extracts a K-dimensional SpotLight sketch $v(\mathcal{G})$ for every $\mathcal{G}$, such that the graphs containing the sudden (dis)appearance of large dense subgraphs are 'far' from 'normal' graphs in the sketch space. In detail, SpotLight sketching first chooses K

query subgraphs $\{(S'_k, D'_k)\}_{k=1}^{K}$ by sampling each source (or destination) into each S'_k (resp. D'_k) with probability p (resp. q).

Second, it exploits the distance gap in the sketch space to detect graphs yielding anomalous sketches as anomalous graphs. A schematic is given in Fig.1.

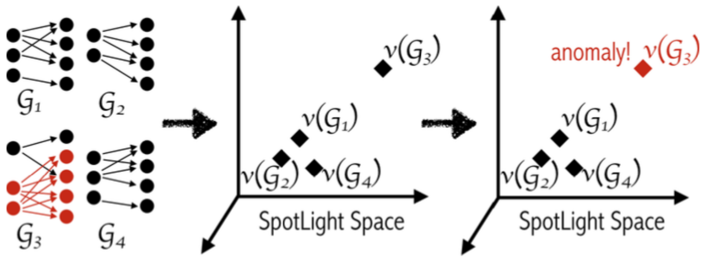


Figure 1: Overview of SpotLight algorithm

3 EXPERIMENTS

3.1 Experimental setup

The experiments were performed on a macOS® Ventura 13.3.1, powered by an Apple® M1 Chip with an 8-Core™ 64-bit processor operating at 3.2 GHz, 7-Core™ GPU, 16-Core™ Neural Engine, 8 GB of unified memory (RAM), and 256 GB SSD storage. The machine learning algorithms were implemented using the Python® programming language and Jupyer Notebook.

3.2 Results and Discussion

Evaluation Metric used include Recall, Precision and AUC (Area Under ROC Curve). The method outputs an anomaly score (higher is anomalous) per graph. Sorting these in descending order, we compute the number of anomalies caught $TP(k)$ (**true positives**) among the top k most anomalous graphs, for every k.

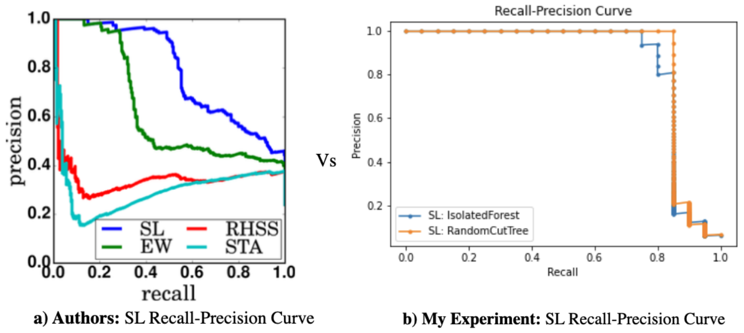


Figure 2: Spotlight Recall-Precision curve

3.3 Result Comparison: (My Result vs the Authors Result)

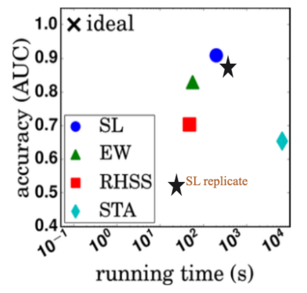
Precision and recall: In Fig. 2, **(a)** compares the recall-precision curve of the methods used in the paper. The original SL model, based on 256 graphs, consistently outperforms all baseline models, with the curve lying completely above. Meanwhile, Fig. 2 **(b)** shows the **recall-precision** curve of my SL model experiments with 730 graphs. I used the Isolated Forest (IF) and the RRCF (Robust Random

Cut Trees) to calculate the anomaly score. The original paper used RRCF to calculate the anomaly score after sketching. In Fig. 2 **(b)**, both approaches perform well on precision and recall but begin to drop slightly towards the end. This could be due to overfitting or data imbalance issues in my data collection.

Methods	AUC (Darpa)
SpotLight	0.91
RHSS	0.83
STA	0.70
SpotLight+IF	0.85
SpotLight+RRCF	0.86

Table 1: AUC Values for Different Methods

a) **My Experiment:** AUC Values for Different Methods



b) **My Experiment:** AUC vs Running Time

Figure 3: AUC score of different Methods vs Running Time

Accuracy vs Running Time In Fig. 3 **(a)**, our experiments show that **SL+IF** has an AUC of 0.85, and **SL-RRCF** (Robust Random Cut Tree) has an AUC of 0.86. This performance is slightly below the original SL paper, which achieved the highest accuracy with **AUC=0.91**. The AUC of SL+IF is 8.4% higher than RHSS (AUC=0.83) and 30% higher than STA [2] (AUC=0.70). However, this gain comes at the cost of a mere 4× slowdown compared to STA [2] and RHSS [3].

Furthermore, Fig. 3 **(b)** shows our SL experiment in comparison to the SL original paper; our AUC result also outperforms the baseline models (RHSS and STA). Additionally, we achieved a high running time of 4.26×10^3 seconds, which is the same as the SL-original paper.

3.4 The Strengths and Weaknesses of Spotlight

The key **strength** of SpotLight [1] lies in its utilization of the randomized sketch algorithm. This feature renders it more scalable, fast, and reliable for identifying the sudden appearance of anomalies in densely directed subgraphs compared to previous models. **Limitation:** However, it still demands a substantial amount of time for implementation, and the time complexity grows as the data streams increase. The SpotLight code presents another limitation in its implementation as the full code is not available to the public, only the algorithm is open source.

4 Conclusion

In Project 2, I successfully implemented the SpotLight [1] algorithm for anomaly detection in streaming graphs with the Darpa dataset and I obtain an AUC score of 0.86 slightly below the original result. **Future work** will address data imbalance and overfitting concerns.

References

[1] Dhivya Eswaran, Christos Faloutsos, Sudipto Guha, and Nina Mishra. Spotlight: Detecting anomalies in streaming graphs. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1378–1386, 2018.

[2] Jimeng Sun, Dacheng Tao, and Christos Faloutsos. Beyond streams and graphs: dynamic tensor analysis. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 374–383, 2006.

[3] Stephen Ranshous, Steve Harenberg, Kshitij Sharma, and Nagiza F Samatova. A scalable approach for outlier detection in edge streams using sketch-based approximations. In *Proceedings of the 2016 SIAM international conference on data mining*, pages 189–197. SIAM, 2016.