GAD : Graph Anomaly Detection, Seminar

Lyes BOURENNANI
Supervisor : Pierre PARREND

Laboratoire de Recherche de l'EPITA

July 3, 2024

Table of Contents

What is Anomaly Detection ?

What is Anomaly Detection ?

- Anomaly detection is a non-supervised analysis which identifies anomalies without prior labeling

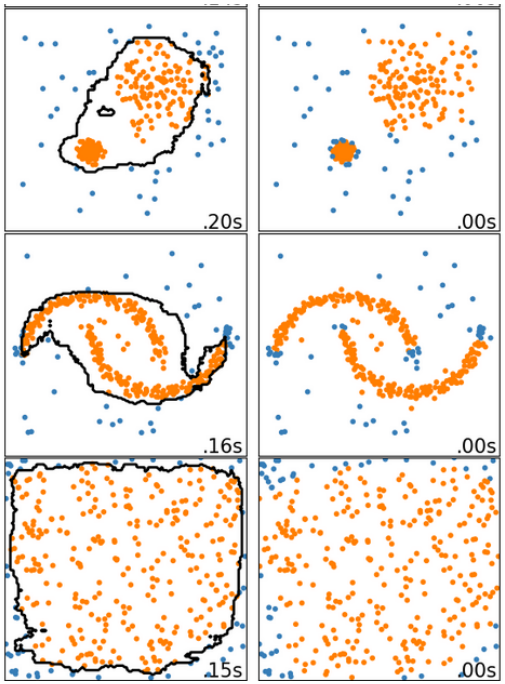


Figure: Isolation Forest vs Local Outlier Factor [5]

Why use Anomaly Detection ?

Problematic

- Machine learning is a viable solution for Anomaly Detection [2]
- Machine Learning is directly applied on data to detect cyberattacks
  $\Longrightarrow$ Missed information on communications between machines

Approach

- Using graphs to represent communications in detection using machine learning
- Apply GAD algorithms to retrieve anomalies information and enhance attack detection

How to use Anomaly Detection ?

Which algorithm and parameters to choose ?

- There exist multiple GAD algorithms
  $\Longrightarrow$ Which one to use ?
- Some algorithms provide hyper parameters and parameters
  $\Longrightarrow$ How to determine the best combination ?
  $\Longrightarrow$ How to improve anomaly detection ?

Approach

- Benchmarking
- Identifying and characterizing key mechanisms
- Make a summary of found results

Types of anomalies and Graph example

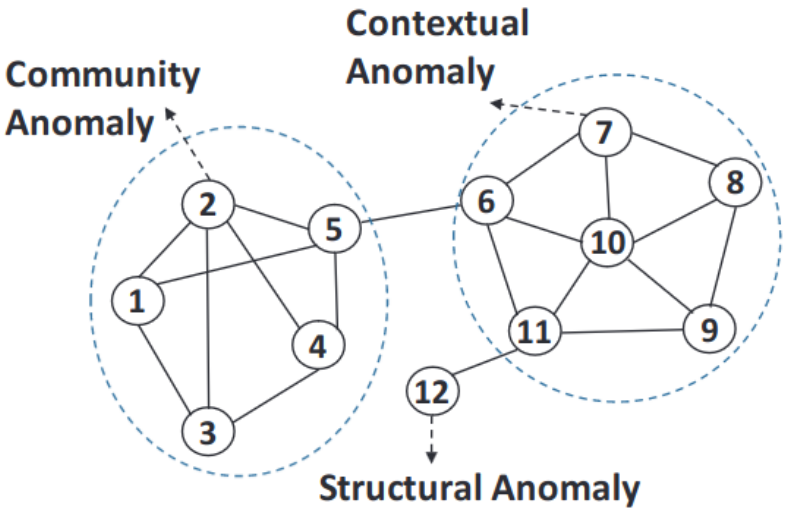
Types of anomalies

- Community anomalies
- Contextual anomalies
- Structural anomalies

Focus for benchmarking

- Structural anomalies are interesting to study the impact of networks structures on attack detection
- It is hard to find datasets with relevant attribute data for anomaly detection

Types of anomalies and Graph example



ID	f1	f2	f3
1	2	22	48
2	11	23	48
3	3	21	47
4	4	22	46
5	3	24	49
6	12	19	51
7	11	38	55
8	10	19	56
9	13	20	49
10	14	20	49
11	11	21	53
12	12	22	52

Figure: [3] Graph example

SCAN [7] (Xu, 2014) Algorithm

Structural **C**lustering **A**lgorithm for **N**etworks [7]

- Uses structure similarity and neighborhood of vertices to find anomalies
- Can be used for clustering
- Depends on two parameters ϵ, μ that act as thresholds on structure similarity and neighborhood

Limits

- The output is a boolean (Not precise enough)
- Only detects structural anomalies

RADAR [3] (Li, 2017) Algorithm

Residual analysis for **A**nomaly **D**etection in **A**ttributed netwo**R**ks [3]

- Designed for attributed networks
- Based on residual analysis (technique used to assess the quality of a model)
- Uses matrix operations
- It uses three parameters (α, β, γ). α defines the row sparsity, β defines the number of anomalies and γ balances the residual analysis information

Limits

- Matrix operations use a lot of memory

ANOMALOUS [6] (Peng, 2018) Algorithm

ANOMALOUS [6]

- Designed for attributed networks
- Based on residual analysis (technique used to assess the quality of a model)
- Uses matrix operations
- Uses CUR decomposition (Matrix approximation method involving matrix product)
- Uses 4 parameters (α, β, γ, ϕ). α and β defines the sparsity. γ and ϕ contributes in balancing the residual analysis.

Limits

- Matrix operations use a lot of memory
- Depends on a lot of parameters (Hard to tune)

Work done on algorithms

Algorithms

- All three algorithms were implemented
- They are in the GPML library

GPML

- Graph Processing library for Machine Learning
- Maintained by Julien MICHEL and Majed JABER (LRE Security and Systems Team)

Missing features

- Documentation and code comments could not be done since the coding style was reviewed

Benchmarking

Benchmarking

- Identify parameters, hyper parameters and their pertinence in extracting information
- Identify the contribution in cyberattacks detection
- Determine detection capacity on abstract and real attack data graphs

Approach

- Unit testing on simple abstract graphs then benchmarking on real data

Unit testing

Unit testing

- Create a set of graphs to easily create unit tests
- The unit tests are inseted into GPML CI

Category	Type of graph
Network Structure	K5
Network Structure	Balanced tree 2-3
Graph anomalies	2 interconnected 10-Star graphs
Graph anomalies	Double K5 with 1 hub
Graph anomalies	K5 + 3-line
Graph anomalies	Double K5 with 1 interconnection of 2 nodes

Figure: Extract of the Graph database [2]

Benchmarking on real data

Classifiers

- Bagging, Decision Tree Random Forest, XGBoost
- The classifier with the best results was XGBoost [1]

Pipeline for real data benchmarking

- Divide the dataset using a time window (5 minutes) and build smaller graphs
- We run the different classifiers with the dataset
- We plot the metrics (Precision, Recall, F1 Score and Balanced accuracy)

Which dataset was used ?

- We used UGR16 [4], a real data dataset containing attacks listed by spanish ISPs

SCAN [7] Benchmarking results on UGR16 - SCAN11

Figure: Metrics comparison between no GAD and SCAN [7] benchmark on SCAN11.

SCAN [7] Benchmarking results on UGR16 - NERISBOTNET

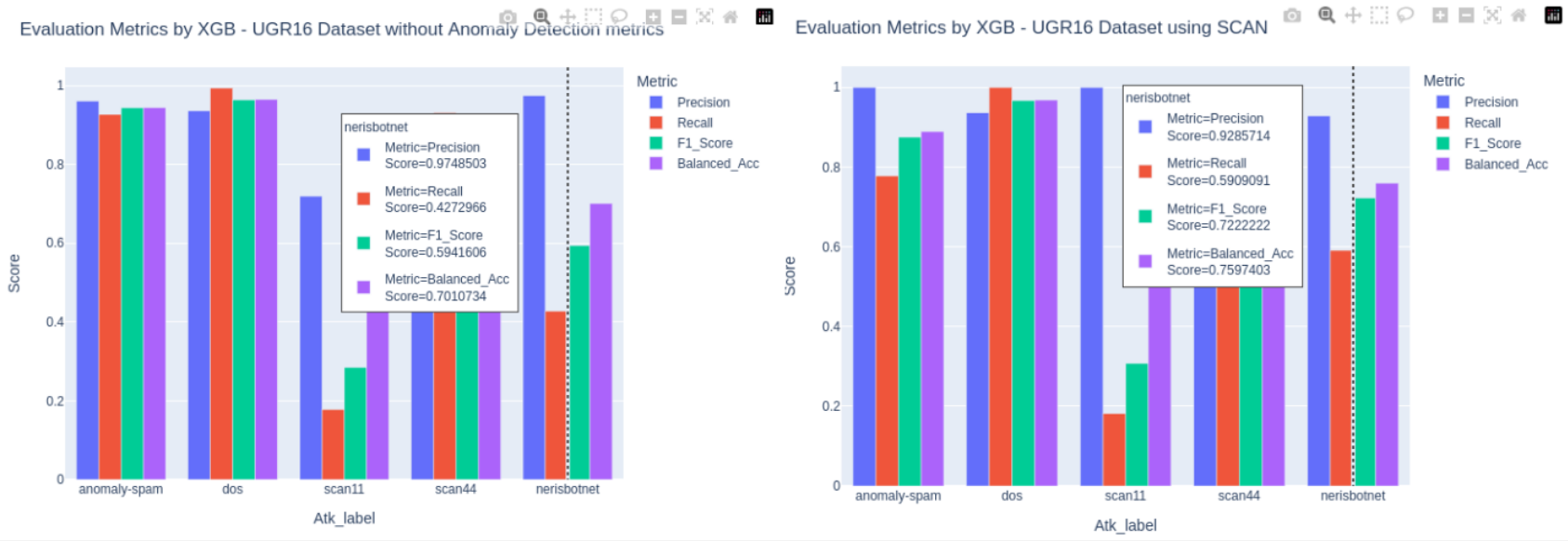


Figure: Metrics comparison between no GAD and SCAN [7] benchmark on NERISBOTNET.

RADAR [3] Benchmarking results on UGR16 [4] - SCAN11

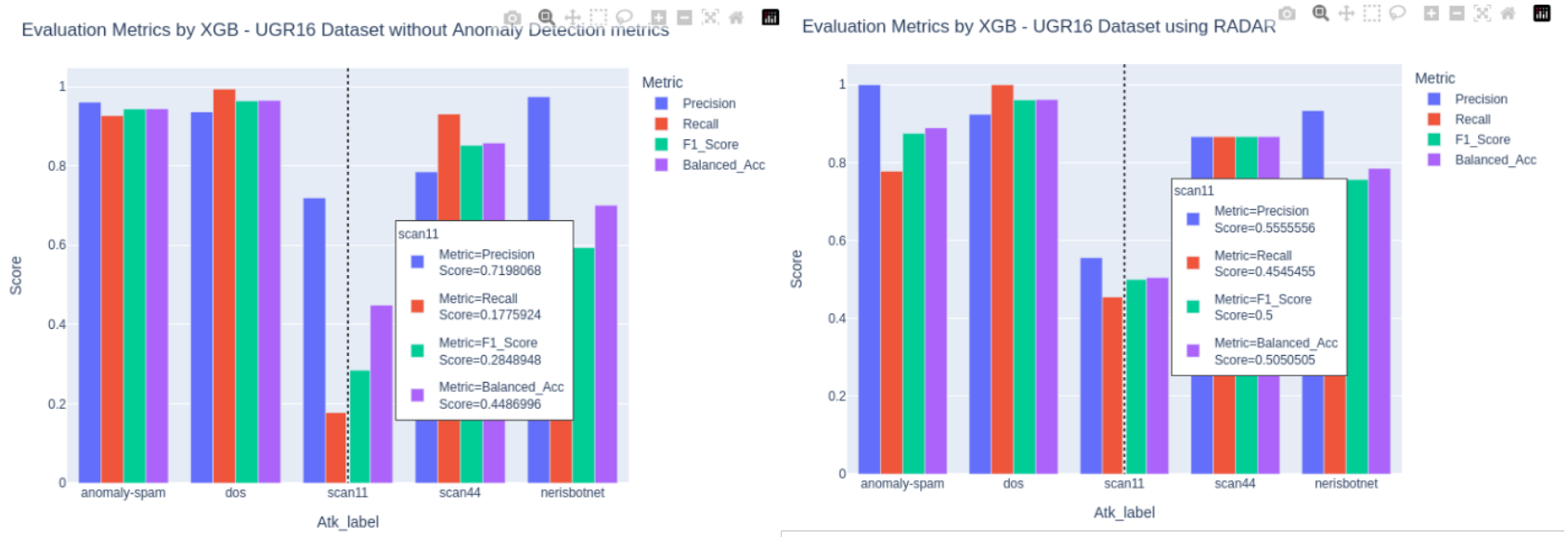


Figure: Metrics comparison between no GAD and RADAR [3] benchmark on SCAN11.

RADAR [3] Benchmarking results on UGR16 [4] - NERISBOTNET

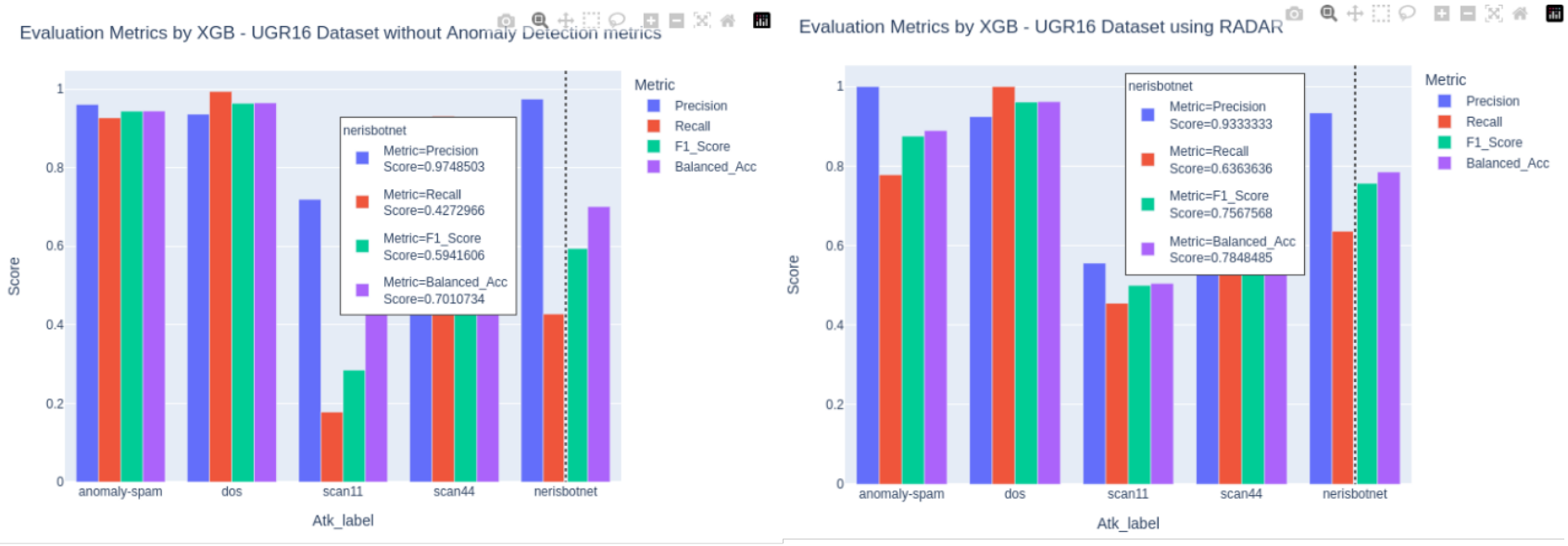


Figure: Metrics comparison between no GAD and RADAR [3] benchmark on NERISBOTNET.

ANOMALOUS [6] Benchmarking results on UGR16 [4] - SCAN11

Figure: Metrics comparison between no GAD and ANOMALOUS [6] benchmark on SCAN11.

Benchmark results on UGR16 [4]

SCAN11

- SCAN [7] adds 15% to the balanced accuracy
- RADAR [3] adds 6% to the balanced accuracy
- ANOMALOUS [6] adds 9% to the balanced accuracy

NERISBOTNET

- SCAN [7] adds 5% to the balanced accuracy
- RADAR [3] adds 8% to the balanced accuracy

What can be seen ?

- GAD Algorithms can contribute in detecting certain type of attacks
- SCAN [7] performs the best on detecting SCAN11
- RADAR [3] performs better on detecting NERISBOTNET

Conclusion on GAD

Conclusion

- GAD algorithms can be pertinent in attack detection
- Some algorithms perform better on specific attack patterns

Next step ?

- Another iteration on real data benchmarking could have given more interessant results
- A better study of the graph database could have given more insights on the impact of network structures

Mistakes and lessons learned

- I spent too much time on developping the algorithms
- I underestimated the importance of the graph database in the early phase of the project

References I

Léo Grinsztajn, Edouard Oyallon, and Gaël Varoquaux.

Why do tree-based models still outperform deep learning on typical tabular data?

Advances in neural information processing systems, 35:507–520, 2022.

Majed Jaber, Nicolas Boutry, and Pierre Parrend.

Towards attack detection in traffic data based on spectral graph analysis.

In *CCE'23, Baku, Azerbaïdjan, mars 2023*, 2023.

Jundong Li, Harsh Dani, Xia Hu, and Huan Liu.

Radar: Residual analysis for anomaly detection in attributed networks.

In *IJCAI*, volume 17, pages 2152–2158, 2017.

References II

Gabriel Maciá-Fernández, José Camacho, Roberto Magán-Carrión, Pedro García-Teodoro, and Roberto Therón.

Ugr '16: A new dataset for the evaluation of cyclostationarity-based network idss.

Computers & Security, 73:411–424, 2018.

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, 12:2825–2830, 2011.

References III

Zhen Peng, Minnan Luo, Jundong Li, Huan Liu, Qinghua Zheng, et al.

Anomalous: A joint modeling approach for anomaly detection on attributed networks.

In *IJCAI*, pages 3513–3519, 2018.

Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas AJ Schweiger.

Scan: a structural clustering algorithm for networks.

In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 824–833, 2007.