Graph Based Anomaly Detection in Chimbuko: Feasible or Fallible?

Chase Phelps[1], Ankur Lahiry[1], Tanzima Z. Islam[1], Christopher Kelly[2]
[1]Texas State University, [2]Brookhaven National Laboratory

Abstract

Chimbuko is a framework for real-time detection of performance anomalies in large-scale workflow applications. Detection of performance anomalies can aid in discovery of algorithmic inefficiencies or potential software/hardware issues in an application's environment. In this study, we investigate the applicability of graph-based deep learning methods for anomaly classification. We hypothesize that transforming tabular performance data into a graph will allow sample correlations to be modelled implicitly, thus allowing graph-based deep learning methods to be build more effective embeddings. We propose to map events to nodes and calculate edge weights using distance between features. Our evaluations show that the proposed method achieves 93% classification accuracy compared to 80% for the state-of-the-art baseline approach, demonstrating graph-based anomaly detection technique is not only feasible, but also beneficial.

Goal

- Investigate Graph-based Representation Learning techniques for improving the accuracy of performance anomaly classification
- Provide insights into the classification model's decision-making process

Background

- Graph Embeddings: A per-node low dimensional representation which combines the node's features with features from neighboring nodes
- Provenance Database: A schema storing metadata about anomalous and normal executions and can be represented conceptually as a table. The figure on the right is an example of the metadata per sample.

Event Information

- **CallStack**
  - **FID**
  - **FUNC**
  - **EXIT**
  - **....**
  - **IS_ANOMALY**
- **PID**
- **COUNTER_EVENTS**
  - **. . .**

Our Approach

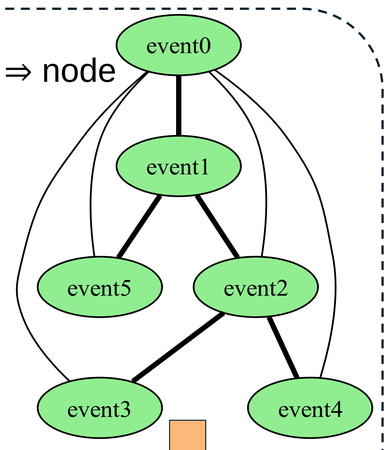
Provenance Database ⇒ table

▸ 3349	18	io_time	TIME	outlier_severity	memin...
0		2218564	1752	1752	504930
1		2218564	43	16	504930
2		2218564	6	6	504930
3		2218564	11	7	504930
4		2218564	4	4	504930
5		2218564	1870	313	504930

Baseline classifier working on the tabular data

Non-graph-based classifier

Event ⇒ node



Our Proposed approach

Graph-based Node Classifier

- Problem formulation: Classify nodes in the graph using their features
- Rationale: Graph edges capture sample correlations explicitly resulting in feature representations (embeddings) that capture salient information. The better the embeddings, the more accurate the classifier.

- Challenge: Performance graph is not given
- Contribution: Methodology for graph construction and modeling

Step 1: Graph construction
- Performance events ⇒ nodes
- Edge weights ⇒ feature similarity
- Label: f: V ⇒ D (anomalous or normal)

Step 2: Modeling
- Graph neural network for representation learning
- 5-fold cross-validation and stratified sampling

Evaluation of Graph-based Anomaly Classification

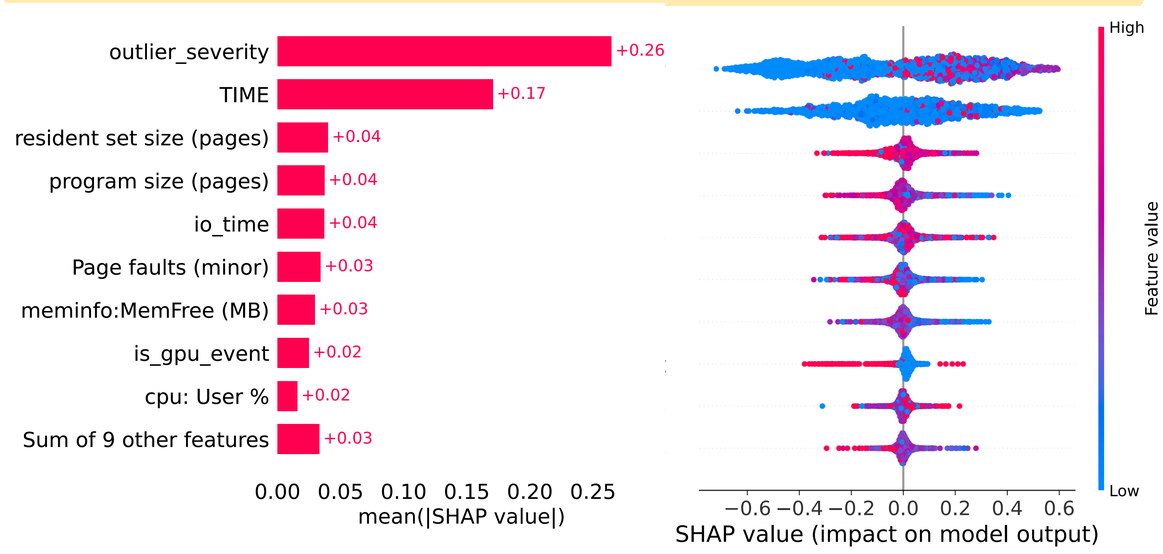
Objective: Evaluate the effectiveness of graph-based techniques compared to the state-of-the-art methods in anomaly classification

	F1	Prec	Recall	Accuracy
GAT	80.42%	80.31%	80.53%	77.86%
GCN	87.11%	86.88%	87.34%	85.44%
SAGE	95.69%	96.06%	95.31%	95.10%
XGBoost (baseline)	82.28%	81.06%	83.57%	79.55%

- Graph based methods outperform XGBoost as XGBoost does not leverage feature relationships across events
- GraphSAGE aggregates a weighted sum of features from k-hop neighbors, while GCN and GAT aggregate features from 1-hop away missing correlations across large number of samples

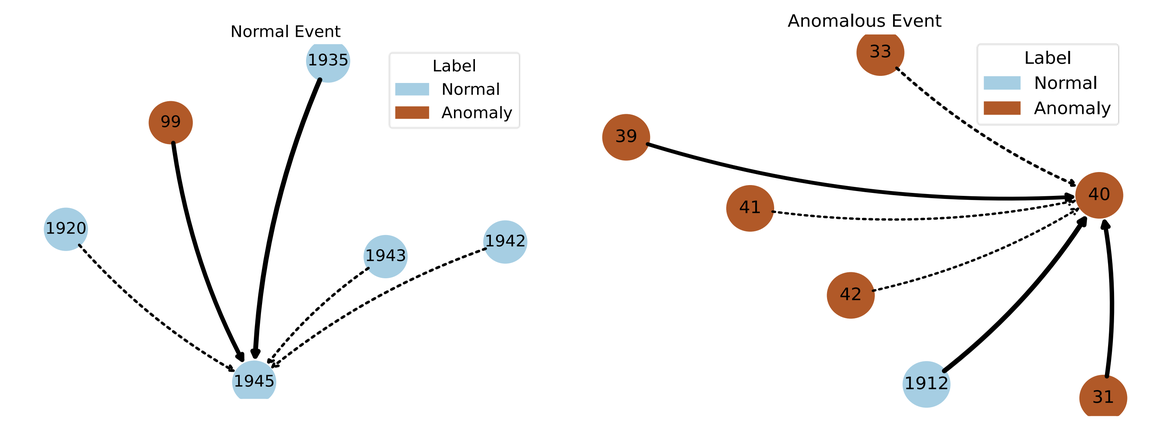
Feature Importance and Model Explanation

Objective: Identify the important features and rank them using feature importance analysis



- There is a higher importance for features related to time and memory
- Time metrics are used to determine when to label an event as anomalous so are expected to correlate with an event's label
- Anomalous events occur when there is less free memory available
- Anomalies occur during high CPU usage

Objective: Explain why a sample was classified as anomalous using the Graph Neural Network-based model



- Edges are weighted by feature similarity. Dotted edges means less influence; solid edges represents higher influence
- Generated embeddings reflect relationships among nodes
- Neighbors of a node tend to share the same label

Future Work

- Use more baselines
- Other distance measures

References

- Kelly, C., et al. (2020). Chimbuko: A workflow-level scalable performance trace analysis tool. In *ISAV'20 In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization* (pp. 15-19).

This material is based upon work supported by the U.S. Department of Energy, Office of Science under Award Number DE-SC0022843 and DE-SC0023173. The work was also supported in part by the Exascale Computing Project (ECP) funded by the U.S. Department of Energy Office of Science. Special thanks to the Sustainable Horizons Institute for the support. Not export controlled.