Investigating Anomalies in Compute Clusters: An Unsupervised Learning Approach

Yiyang Lu†, Jie Ren†, Yasir Alanazi§, Ahmed Mohammed§, Diana McSpadden§, Laura Hild§, Mark Jones§, Wesley Moore§, Malachi Schram§, Bryan Hess§, Evgenia Smirni†
† William & Mary, § Thomas Jefferson National Accelerator Facility

ABSTRACT

As compute clusters continue to grow in scale and complexity, the frequency of detected anomalies in their operation significantly increases. Timely detection of anomalous events is vital to maintain system efficiency and availability. This study presents an attention-based graph neural network (GNN) for detecting anomalies in clusters at the compute node level and for providing detailed root cause analysis. We show the effectiveness of attention-based GNNs to accurately detect and localize anomalies on real-world datasets.

1 INTRODUCTION

Anomalies in a compute cluster are unexpected deviations in the operation of hardware componentsn that cause performance degradation, system instability, or even complete failure of components or the entire cluster [3, 4, 12, 14, 18]. Detecting and diagnosing such anomalies in heterogeneous clusters with batch jobs is challenging, even for small scale clusters. In such environments, automatic anomaly detection and root cause analysis contributes to better system operation with timely detection of an anomaly and its resolution.

Detecting and diagnosing anomalies in compute clusters often requires periodically collecting hardware and software metrics, typically in the form of time series. Metrics collected from CPU, memory, and disk can indicate the status of a single compute node. In addition, metrics collected from the network, workflow management tools such as Slurm[17], and network-shared file systems such as Lustre [13] can point to anomalies that may propagate to compute nodes executing the same job. This work focuses on single-node anomaly prediction using CPU, memory, and disk information in a cluster setting. **E: no network info?**

Deep Learning (DL) is effective in detecting anomalies in complex environments [2, 5, 8–10]. In a compute cluster environment, there are several challenges in implementing DL-based anomaly detection. First, labeling anomalies is nearly impossible, since system administrators identify them not by standardized thresholds for specific metrics [15, 16], but by the overall behavior of the entire compute node. Additionally, anomalies are rare in compute clusters, with only 0.035% of all real-world cluster events classified as anomalies [6]. To address this challenge, we use *unsupervised learning* to train our detection model with data on normal node operation.

The second challenge arises from the high-dimensional, multivariant nature of the data. Existing DL-based anomaly detection rely on autoencoders (AE), requiring an understanding of specific monitored metrics that are pivotal in pinpointing anomalies. These metrics are often tightly coupled and their significance in detecting anomalies may dynamically shift depending on the anomaly source [7]. AE-based solutions fail to capture the implicit relations between monitored metrics within a cluster setting. To address this challenge, we build relationship graphs for all monitored metrics and employ an attention-based mechanism, see Section 2.

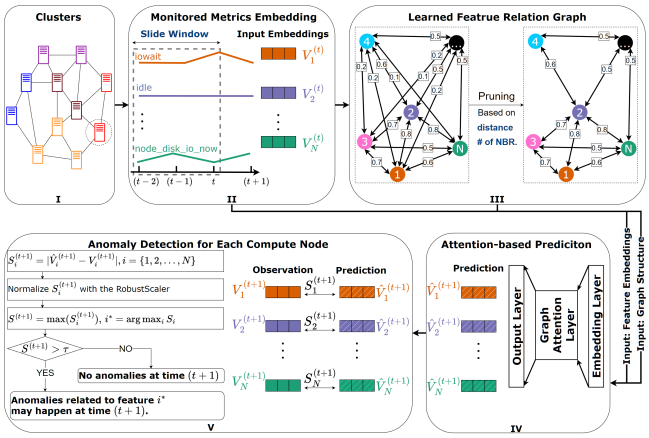

Figure 1: Workflow of attention-based GNN for anomaly prediction.

Last but not least, the anomaly detection solution should apply to different hardware metrics but what is definied as anomaly may be different across different hardware components. Here, we maintain the generality of the DL model while customizing the anomaly identification process to adapt to different hardware metrics.

2 DESIGN

We propose an anomaly detection framework using an attention-based GNN that is trained in an unsupervised manner, i.e., using only data collected while the system is not experiencing an anomaly. Figure 1 provides the workflow of the framework, (I) data collection, (II) monitored metrics embedding, (III) learned features relation graph, (IV) attention-based GNN model, and (V) anomaly detection for each compute node. Specifically, data collection (I) and GNN model training (II-IV) happen offline, and anomaly detection (V) happens online. We introduce each of the components below:

Monitored metrics embedding. Embeddings built over the historical time series data capture the unique characteristics of each monitored metric. Input embeddings for each monitored metrics are organized as vectors with w elements, where w is a hyperparameter representing the size of the sliding window (time).

Relation graph learning. A directed graph is created using monitored metrics embeddings. Nodes represent monitored metrics, while edges show dependency relationships between them. We use the cosine distance is used to calculate edge weights, then prune the graph based on the distance (*EdgeThreshold*) and number of neighbors (*topK*), where *topK* limits the number of features contributing to a single feature by selecting only the top 'k' features for

each node. *EdgeThreshold* refines the graph structure by preserving only those edges with a similarity exceeding the set threshold.. Component III in figure 1 demonstrates a learned relation graph with *EdgeThreshold* and *topK* set to 0.5 and 5, respectively.

Attention-based prediction. We predict each monitored metric at each time step with, utilizing the past behavior of the monitored metric and its neighboring metrics within the learned relation graph of the GNN. The relation graph, once learned, enables capturing of all monitored metrics associated with anomalies.

Anomaly detection for each compute node. As show in component V in figure 1, we identify anomalies as significant deviations from the ground truth (i.e., the expected behavior). Our model detects and explains anomalies by computing individual scores S_i^t for each monitored metric i at time step t. To localize anomalies, we perform a robust normalization of each monitored metric to prevent potentially overly dominant deviations of a single metric. The framework further computes the overall anomaly scores S^t by aggregating all individual scores with the max function. Anomalies are detected when the overall anomaly scores exceed a threshold τ.

3 EVALUATION

Experiment setup. We evaluate our anomaly detection framework on a scientific computing cluster within the Jefferson Lab production environment. The cluster has 332 computational nodes divided into five groups with different hardware properties. All monitored metrics from each compute node are consolidated into one single monitor node, facilitated by the Prometheus database. We focused on CPU, memory, and IO anomalies. The datasets included 8 CPU metrics, 47 memory metrics, and 11 disk metrics, comprising over one million records with 181GB raw data. Leveraging an automatic hyperparameter tuner[1], we choose a combination of w, *EdgeThreshold*, and *topK*, which performed best on a validation dataset achieving minimum anomaly scores for each input feature.

Efficiency in detecting synthetic anomalies. To verify the efficiency of the anomaly detection framework, we inject Gaussian noise into different monitored metrics and create a synthetic dataset with anomalies. We use the framework to find (1) Which compute node is abnormal? (2) Which hardware component and its corresponding feature is responsible for the (predicted) anomaly?

We evaluate five compute node groups with different hardware properties, and set the anomaly threshold τ_1 and τ_2 as the 100^{th} percentile and the 99.99^{th} percentile of the normal data, respectively. Consistent with standard practice, we report the precision of the anomaly prediction at the compute node defined as the fraction of actual anomalies among the predicted anomalies [11].

To quantify the answer to the second question, we define the accuracy of root cause analysis, which is calculated as the ratio of successful root cause identifications to the predicted anomalies. Table 1 shows the results. Our framework successful detects 89% of anomalies in five groups of compute nodes. Specifically, our framework is more sensitive to CPU and memory anomalies, and pinpoints the root cause of anomalies 86% on average.

Efficiency in detecting anomalies on real-world cluster. We tested our framework using the disk anomalies dataset from Jefferson Lab. We observe that the attention-based GNN model can accurately predict monitored metrics' value with mean squared error (MSE) of only 0.001. Moreover, the model accurately detects anomalies, including nodes that lack a clearly anomalous signature but are identified as anomalous in ground truth.

Table 1: Evaluating on synthetic anomalies. "Pre" is short for "precision" and "RCA" is short for "root cause accuracy".

			G14	G16	G18	G19	G23	Avg.
CPU	Pre.	τ_1	0.72	0.72	0.92	1	1	0.87
		τ_2	1	1	1	1	1	1
	RCA	τ_1	0.68	0.56	0.92	1	1	0.83
		τ_2	1	1	0.875	1	1	0.98
Memory	Pre.	τ_1	0.74	0.86	1	1	1	0.92
		τ_2	1	1	0	1	1	0.8
	RCA	τ_1	0.74	0.82	1	1	1	0.91
		τ_2	1	1	0	1	1	0.8
Disk	Pre.	τ_1	0.63	0.79	1	1	1	0.88
		τ_2	0	1	0	1	1	0.6
	RCA	τ_1	0.47	0.75	1	1	1	0.84
		τ_2	0	1	0	1	1	0.6

4 CONCLUSION

In this work, we propose an unsupervised attention-based GNN that learns a graph of relationships between monitored metrics to detect deviations and provide root cause analysis. Our approach is validated using real-world datasets, demonstrating its capability in accurately detecting and localizing anomalies. Our future work aims to explore additional real-world anomalies and enable continual learning with GNN to detect anomalies in complex, dynamic compute clusters.

ACKNOWLEDGEMENT

Yiyang Lu and Evgenia Smirni are partially supported by NSF grant 2130681. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Nuclear Physics under contract DE-AC05-06OR23177.

REFERENCES

[1] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A Next-generation Hyperparameter Optimization Framework. http://arxiv.org/abs/1907.10902 arXiv:1907.10902 [cs, stat].

[2] Jacob Alter, Ji Xue, Alma Dimnaku, and Evgenia Smirni. 2019. SSD failures in the field: symptoms, causes, and prediction models. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis, SC 2019, Denver, Colorado, USA, November 17-19, 2019*, Michela Taufer, Pavan Balaji, and Antonio J. Peña (Eds.). ACM, 75:1–75:14. https://doi.org/10.1145/3295500.3356172

[3] Ludmila Cherkasova, Kivanc Ozonat, Ningfang Mi, Julie Symons, and Evgenia Smirni. 2008. Anomaly? application change? or workload change? towards automated detection of application performance anomaly and change. In *2008 IEEE International Conference on Dependable Systems and Networks With FTCS and DCC (DSN)*. 452–461. https://doi.org/10.1109/DSN.2008.4630116 ISSN: 2158-3927.

[4] Ludmila Cherkasova, Kivanc Ozonat, Ningfang Mi, Julie Symons, and Evgenia Smirni. 2009. Automated anomaly detection and performance modeling of enterprise applications. *ACM Transactions on Computer Systems* 27, 3 (Nov. 2009), 6:1–6:32. https://doi.org/10.1145/1629087.1629089

[5] Ailin Deng and Bryan Hooi. 2021. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 35. 4027–4035.

[6] Martin Molan, Andrea Borghesi, Daniele Cesarini, Luca Benini, and Andrea Bartolini. 2023. RUAD: Unsupervised Anomaly Detection in HPC Systems. *Future Gener. Comput. Syst.* 141, C (apr 2023), 542–554. https://doi.org/10.1016/j.future.2022.12.001

[7] Bin Nie, Jianwu Xu, Jacob Alter, Haifeng Chen, and Evgenia Smirni. 2020. Mining Multivariate Discrete Event Sequences for Knowledge Discovery and Anomaly Detection. In *50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2020, Valencia, Spain, June 29 - July 2, 2020*. IEEE, 552–563. https://doi.org/10.1109/DSN48063.2020.00067

[8] Bin Nie, Ji Xue, Saurabh Gupta, Christian Engelmann, Evgenia Smirni, and Devesh Tiwari. 2017. Characterizing Temperature, Power, and Soft-Error Behaviors in Data Center Systems: Insights, Challenges, and Opportunities. In *25th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, MASCOTS 2017, Banff, AB, Canada, September 20-22, 2017*. IEEE Computer Society, 22–31. https://doi.org/10.1109/MASCOTS.2017.12

[9] Bin Nie, Ji Xue, Saurabh Gupta, Tirthak Patel, Christian Engelmann, Evgenia Smirni, and Devesh Tiwari. 2018. Machine Learning Models for GPU Error Prediction in a Large Scale HPC System. In *48th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2018, Luxembourg City, Luxembourg, June 25-28, 2018*. IEEE Computer Society, 95–106. https://doi.org/10.1109/DSN.2018.00022

[10] Riccardo Pinciroli, Lishan Yang, Jacob Alter, and Evgenia Smirni. 2023. Lifespan and Failures of SSDs and HDDs: Similarities, Differences, and Prediction Models. *IEEE Trans. Dependable Secur. Comput.* 20, 1 (2023), 256–272. https://doi.org/10.1109/TDSC.2021.3131571

[11] David MW Powers. 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. *arXiv preprint arXiv:2010.16061* (2020).

[12] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. 2022. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment* 15, 9 (May 2022), 1779–1797. https://doi.org/10.14778/3538598.3538602

[13] Philip Schwan et al. 2003. Lustre: Building a file system for 1000-node clusters. In *Proceedings of the 2003 Linux symposium*, Vol. 2003. 380–386.

[14] Ozan Tuncer, Emre Ates, Yijia Zhang, Ata Turk, Jim Brandt, Vitus J. Leung, Manuel Egele, and Ayse K. Coskun. 2019. Online Diagnosis of Performance Variation in HPC Systems Using Machine Learning. *IEEE Transactions on Parallel and Distributed Systems* 30, 4 (April 2019), 883–896. https://doi.org/10.1109/TPDS.2018.2870403 Conference Name: IEEE Transactions on Parallel and Distributed Systems.

[15] Ji Xue, Robert Birke, Lydia Y. Chen, and Evgenia Smirni. 2016. Tale of Tails: Anomaly Avoidance in Data Centers. In *35th IEEE Symposium on Reliable Distributed Systems, SRDS 2016, Budapest, Hungary, September 26-29, 2016*. IEEE Computer Society, 91–100. https://doi.org/10.1109/SRDS.2016.021

[16] Ji Xue, Robert Birke, Lydia Y. Chen, and Evgenia Smirni. 2018. Spatial-Temporal Prediction Models for Active Ticket Managing in Data Centers. *IEEE Trans. Netw. Serv. Manag.* 15, 1 (2018), 39–52. https://doi.org/10.1109/TNSM.2018.2794409

[17] Andy B Yoo, Morris A Jette, and Mark Grondona. 2003. Slurm: Simple linux utility for resource management. In *Workshop on job scheduling strategies for parallel processing*. Springer, 44–60.

[18] Qi Zhang, Ludmila Cherkasova, Guy Mathews, Wayne Greene, and Evgenia Smirni. 2007. R-Capriccio: A Capacity Planning and Anomaly Detection Tool for Enterprise Services with Live Workloads. In *Middleware 2007 (Lecture Notes in Computer Science)*, Renato Cerqueira and Roy H. Campbell (Eds.). Springer, Berlin, Heidelberg, 244–265. https://doi.org/10.1007/978-3-540-76778-7_13