BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

EXTENDING GRAPH MACHINE LEARNING FOR INTERPRETABLE MULTI-CLOUD INCIDENT PREDICTION AND LIFECYCLE-AWARE ROOT CAUSE ANALYSIS : *A Review and Proposed Graph-Based Framework*

[1]Atharva Kadam, [2]Prof. Sandeep Kumar Jaiswal, [3]Prof. Arvind Gautam

[1]Student, [2]Assistant Professor and Guide, [3]Seminar Coordinator
[1]Department of Artificial Intelligence and Data Science,
[1]Ajeenkya DY Patil School Of Engineering, Pune, India

Abstract

Many applications span multiple cloud providers. The faults in one cloud provider can travel to other providers, which are not visible in the monitoring consoles of those providers. Threshold alerts and anomaly detection systems can only monitor each metric individually and therefore are often alerted too late to diagnose the problem quickly. This article reviews the application of graph machine learning techniques, specifically graph neural networks (GNNs), to incident prediction and root cause analysis and extends these techniques to multi-cloud environments. The metrics from each cloud provider are combined into a graph structure and GNNs learn from that graph structure. The results of the analysis can be explained by returning the subgraph of the graph that is most responsible for the results of the analysis. These results can be applied throughout the incident lifecycle. Also, metrics for evaluating the success of these systems are specified. This work is a proposal for such a system and has not been validated through new experiments.

Keywords: Graph machine learning, Graph neural networks, Multi-cloud, Incident prediction, Root cause analysis, Interpretability, AIOps

Introduction

Cloud computing allows companies to run their workloads on multiple clouds simultaneously. This allows them to avoid being dependent on a single cloud vendor, to comply with regulatory requirements or to ensure that their applications remain available in the event of an outage of the cloud vendor (Beyer et al., 2016).

The process of dealing with an incident can be broken down into several steps. However, tooling is often limited to addressing just one of these steps. For example, alerts that can be triggered when certain thresholds are reached will typically do so only after the customers have already been impacted by the

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

incident and root cause analysis is typically performed from a starting point of zero with the clock still running (Luo et al., 2014; Chen et al., 2019).

Cloud computing can be represented as a graph, with services connected to one another, running on hosts and containers, located in specific regions and using databases and queues. Graph neural networks learn about the structure of these graphs (Kipf & Welling, 2017; Gilmer et al., 2017), which would allow them to determine the potential propagation of issues within a cloud environment. However, graph neural networks do have some drawbacks in the setting of cloud computing, specifically in the area of credibility: it is unlikely that engineers would take actions based upon the output of a graph neural network if they could not explain that output (Rudin, 2019).

We therefore survey graph-based work on incident prediction and RCA, point out what multi-cloud operation still lacks, and outline a framework that is interpretable and lifecycle-aware from the start. Our contribution is a structured review, a framework and an evaluation plan; we do not claim new experimental findings.

Literature Review

Theoretical Framework

Three concepts support the framework. Message passing is the first: every node refreshes its representation from its neighbours, so the structure of the system becomes part of what the model learns (Gilmer et al., 2017). Graph convolution, neighbour sampling and neighbour attention are common forms of it (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018), and reviews of the area are available (Zhou et al., 2020; Wu et al., 2021).

The second concept is that real graphs are typed and change over time. A cloud graph mixes many kinds of nodes and links, which relational and heterogeneous models treat with separate parameters per type (Schlichtkrull et al., 2018; Hu et al., 2020). Because telemetry keeps arriving, dynamic-graph methods add memory or time encodings (Rossi et al., 2020; Kazemi et al., 2020).

The third concept is the link between explanation and causality. Explainers for GNNs point to the subgraph and features that lead to an output (Ying et al., 2019; Luo et al., 2020), while causal reasoning seeks to determine what influences what else in the system (Pearl, 2009). Because RCA aims to find a cause, not just a correlated symptom, it requires both forms of reasoning.

Incident Management and Multi-Cloud Challenges

Table 1 provides a summary of the various stages of the cloud lifecycle and the tasks that are required of those stages, as well as the contributions that graph learning can make to each of these tasks. The three main difficulties that arise when operating across multiple cloud platforms include telemetry differences between the platforms, behaviour differences between the platforms and privacy rules or contracts that do not allow for the combination of data from multiple platforms into one location.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Table 1: Incident Lifecycle Stages and the Role of Graph Learning

Stage	Task	Role of Graph Learning
Pre-incident	Incident prediction	Detect spreading degradation early
Detection	Anomaly alerting	Group related alerts by topology
Triage	Severity, ownership	Locate the affected subgraph
Diagnosis	Root cause analysis	Rank candidate causes along edges
Review	Post-incident learning	Store incident subgraphs for reuse

Classical and Statistical Approaches

Research in early diagnosis of software systems has employed machine learning algorithms to match events with time series data (Luo et al., 2014) and deep learning models to identify abnormal system behavior from log files (Du et al., 2017). However, these approaches analyze individual signals or services within a system without considering the dependencies between those services. Also, these methods tend to degrade in performance when the workload of the system shifts over time, which is known as concept drift (Gama et al., 2014).

Graph-Based Root Cause Analysis

Another body of research utilizes dependency graphs directly. The CauseInfer tool creates a causality graph from the performance data of a system and walks that graph to determine the causes of performance issues (Chen et al., 2014). Tools like MicroRCA and AutoMAP use these dependency graphs to assign scores to services within a system (Wu et al., 2020; Ma et al., 2020). MicroRank analyzes these dependency graphs using range analysis (Yu et al., 2021). Nezha is another tool that can handle multiple data types within a system and provides interpretable results for those analyses (Yu et al., 2023). These tools are more effective than analyzing a single signal from a system but they rely upon the rules set by the users rather than any learning models of the system.

Graph Neural Networks for Anomaly and Incident Prediction

Learned graph models have also been applied to multivariate monitoring data. For instance, graph deviation networks can be used to model the relationships between various metrics in a dataset and report when those relationships are violated (Deng & Hooi, 2021). Graph-attention models can be used to model both the features of each sample in the dataset, as well as the time relationships between those samples (Zhao et al., 2020). These models can be used to identify potential incidents within the monitored system but they do not account for multiple providers of the same type of data. Knowledge graphs, however, can be used to model various entities within the same dataset in a consistent manner (Hogan et al., 2021).

Interpretability

In operations, explanations often decide whether a model gets used. Model-agnostic methods such as LIME and SHAP account for single predictions (Ribeiro et al., 2016; Lundberg & Lee, 2017), and GNN explainers report the edges and features that mattered (Ying et al., 2019; Luo et al., 2020; Yuan et al., 2023). Doshi-Velez and Kim (2017) propose judging explanations against the purpose they serve, and

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Rudin (2019) warns that explaining an opaque model can be less sensible than using one that is interpretable in the first place.

Research Gaps

The review exposes four gaps. Most graph methods assume one environment and not several clouds with mismatched schemas. Prediction and RCA are usually separate systems, so what is learned in one stage is not reused in the next. Explanations are seldom tested for usefulness to responders. And transfer between providers, or learning under data-sharing restrictions, is still largely unexplored.

Research Questions

The following research questions are addressed in this paper:

1. How well do graph-based models compare to non-graph and rule-propagation approaches for early incident prediction and root cause ranking?
2. Which graph design and training choices are needed for models to work across heterogeneous cloud providers?
3. How should explanations be generated and assessed so that they help engineers during triage and diagnosis?

Hypotheses

Three null hypotheses were formulated for the framework:

1. At equal false-alarm rates, graph-based models do not predict incidents earlier or more accurately than non-graph time-series baselines.
2. Heterogeneous temporal graph models are not better than homogeneous or static graph models for root cause ranking.
3. Explanatory subgraphs do not save engineers time in finding a root cause compared to ranked lists without explanation.

Methodology

Research Design

The study is designed with design-science. A structured review of incident management, graph learning and interpretability provides a set of requirements that inform the proposed framework and evaluation plan. Mass testing is postponed to the next stage.

Data Sources and Sample

This framework is based on four types of data: metrics, logs, distributed traces, and change/deployment events. The traces and configurations contain the dependency edges, while the logs and changes contain the event nodes and their attributes. Tickets and post-mortem reports serve as the labels for incident prediction and root cause analysis. In cases when sharing production data is impossible, labelled incidents may be provided by benchmarks and fault injection testbeds of public microservices. Splitting

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

has to be done along the time and incident boundaries to ensure that the related incidents do not split between train and test sets.

Framework Components

The pipeline from Figure 1 consists of seven steps and comprises four main modules:

4. **Unification layer:** translates the provider-specific ids, metric and log names into a common schema to make the same objects look the same in every cloud.
5. **Graph construction:** creates a heterogeneous temporal graph whose nodes have types (service, host, container, region, cloud, change) and whose edges have types (calls, runs-on, shares, deployed-to). These graphs are updated using sliding windows.
6. **Learning module:** a relation-temporal GNN encoder (Schlichtkrull et al., 2018; Hu et al., 2020; Rossi et al., 2020) feeding two heads, one for predicting incidents and one for ranking root causes. Focal loss (Lin et al., 2017) handles infrequent incidents, and domain adversarial training (Ganin et al., 2016) trains the encoder on features independent of the provider.
7. **Interpretability layer:** outputs an explanatory subgraph with edge importance (Ying et al., 2019; Luo et al., 2020) and counterfactual validation, and associates each interpretation with a stage in the lifecycle.

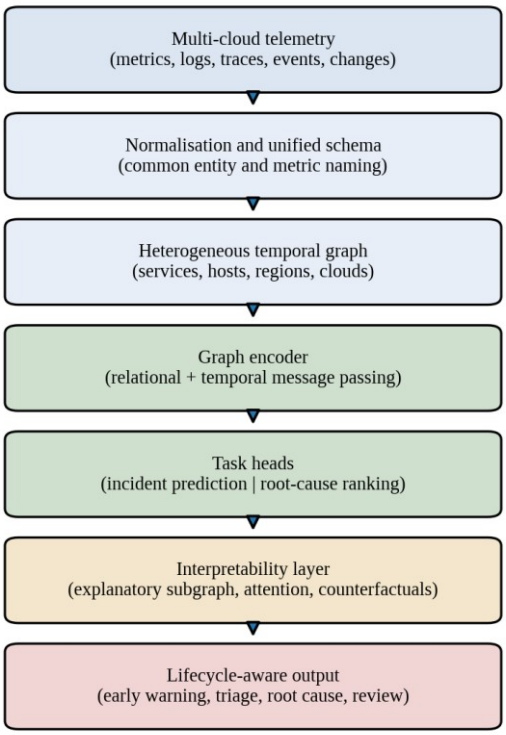


Figure 1: Proposed interpretability and lifecycle-aware multicloud framework

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Experiment Protocol

Evaluation proceeds in four stages:

Phase 1: Data Preparation. Normalize telemetry data, create one graph per time window, and split data according to time and incidents.

Phase 2: Training. Pretrain the encoder with unlabelled windows, and fine-tune both heads When data cannot be transported out of a provider's cloud, federated learning (McMahan et al., 2017) takes place of pooling**.**

Phase 3. Ablate. Remove one component at a time and record the effect:

- No temporal encoding
- No type-specific (heterogeneous) parameters
- No domain-adversarial alignment
- No interpretability layer

Phase 4. Test on clouds and humans. Learn on one cloud vendor and test on another, and measure diagnosis time of engineers using and not using explanations. Figure 2 is an example of expected outcome on a simulated dependency graph.

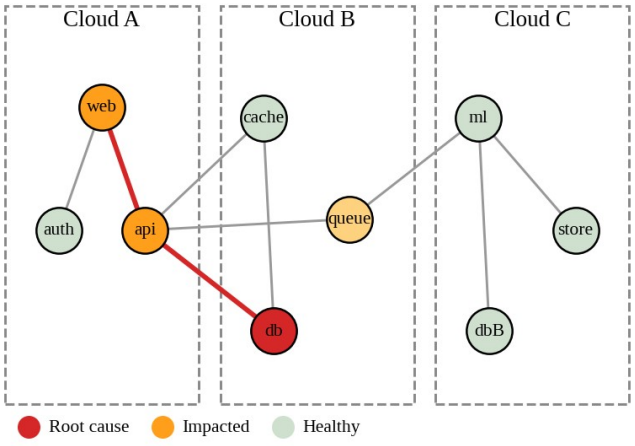


Figure 2: Multi-cloud dependency graph with root-cause path marked out (exemplary) Evaluation Metrics and Analysis

The prediction quality will be measured by Precision, Recall and Lead Time; Root Cause ranking by top-k accuracy and mean reciprocal rank; explanations by fidelity and user study, like in DoshiVelez and Kim (2017). The measurements are summarized in Table 2. The results must be measured in repeated experiments and then evaluated by paired significance tests.

Table 2: Evaluation Metrics

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Metric	Description
Precision / Recall	Correct and found incident predictions
Lead time	Minutes of warning before the incident
False alarms/day	Spurious warnings per day
Top-k accuracy	True cause within the top k candidates
MRR	Mean reciprocal rank of the true cause
Fidelity	How well the explanation reflects the model
Cross-cloud drop	Performance loss on an unseen provider
Diagnosis time	Engineer time to identify the cause

Results

No new experiments are reported, so the results here are analytical. They set the published characteristics of each approach family against what multi-cloud operation demands.

Research Question 1: Graph Models Compared with Other Approaches

Table 3: Comparison of Approach Families

Family	Uses links	Interpretable	Multi-cloud ready
Rules and thresholds	No	High	Low
Per-signal ML	No	Medium	Low
Graph propagation	Yes	High	Medium
GNN (proposed)	Yes	Medium, with explainers	Medium to high

As seen from Table 3, graph propagation-based approaches take into account dependencies and are easily explainable, but have pre-defined rules. Graph Neural Networks learn the dependencies and can work with more advanced data, but require explanations. Signal-based approaches remain simple and ignore fault propagation between services.

Research Question 2: Making Models Work Across Cloud

There are three main technologies which support graph-based learning. First is a common schema to make nodes and edges comparable. Another is having type-aware parameters to prevent edges from merging different nodes. Last is the capability of domain adaptation which decreases a system's provider-specific bias.

Research Question 3: Explanations

Explaining a result for a system that uses a dependency graph should be done with a subgraph that is easily understandable. For example, it should be understandable enough for the operator to act upon it to fix a system fault.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Hypothesis Assessment

Table 4: Assessment of Null Hypotheses at the Framework Level

Hypothesis	Basis of assessment	Status
H_o1	Failures spread along dependencies	Likely rejected; to be tested
H_o2	Typed, time-aware graphs retain more signal	Likely rejected; to be tested
H_o3	Explanations narrow the search space	Likely rejected; to be tested

Table 4 gives the current status of each hypothesis. All are supported by related work, but not tested against a standardized set of cloud benchmarks, so each is stated as likely to be rejected.

Discussion

From this study, I have learned that no technique addresses the complete lifecycle of a fault. However, using graph learning provides benefits in detecting faults early and when the cause of a fault is hard to determine.

Prediction and Diagnosis Together

One common graph representation would allow evidence gathered before an incident to be used during diagnosis and for the results of diagnosis to be used later in predicting incidents. Lifecycle perspective is the key difference compared to tools designed for one phase of the lifecycle.

Interpretability and Trust

Explanations have significance only when they can lead to action. A concise subgraph associated with specific services and recent changes is better than a lengthy feature vector. An explanation must have an element of uncertainty, and if a simple model works as well as a complex one, the former should be favored (Rudin, 2019).

Data Quality and Drift

The models in the cited paper were designed for static settings. The systems, however, are constantly changing due to the continuous deployment and scaling of cloud infrastructure and services. These systems should, therefore, account for some amount of drift in the data (Gama et al., 2014). Along with the previously mentioned factors, root cause identification is subjective. In the proposed models, this adds uncertainty and lowers the accuracy.

Limitations

Although this is not an experimental study, it presents a number of new design considerations for using graph machine learning to help in incident prediction and root cause analysis in a cloud environment.

Conclusion

A number of new approaches were introduced and framed as a system for multiple clouds. Telemetry data from multiple sources was used to build a directed, multigraph representation of a cloud system over time.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

Relational and temporal data were used as inputs to a deep neural network, which was trained to learn both cross-domain (inter-cloud) and domain-specific (intra-cloud) patterns for better diagnosis of events.

Subgraphs are generated for the lifecycle of a particular incident. These provide a better understanding of how a particular event occurred and what processes, events, and other telemetry related to it.

Although it remains an empirical question to confirm that engineers trust and utilize this framework for the intended goal of providing an earlier and more accurate diagnosis, this work builds upon current approaches for a more credible system. This framework has not yet been benchmarked on a cloud-based system

Recommendations

The priorities for future work mentioned in the review include:

8. Create and make open access datasets of labeled multi-cloud incidents. These datasets should contain traces, logs, changes and have labels of the root cause of the incident.
9. Create a standard telemetry format for all service providers. This allows cross comparison of services between service providers.
10. Prediction methods should not be evaluated on accuracy, but on how early the prediction can be made while maintaining a fixed false alarm rate.
11. Prediction models should be evaluated by asking engineers to provide diagnoses and tracking the time it took to do so.

5._How models can be transferred between providers should be evaluated. Approaches to enable models to learn across data that cannot be combined should also be evaluated.

References

1. Beyer, B., Jones, C., Petoff, J., & Murphy, N. R. (Eds.). (2016). Site reliability engineering: How Google runs production systems. O'Reilly Media.
2. Chen, P., Qi, Y., Zheng, P., & Hou, D. (2014). CauseInfer: Automatic and distributed performance diagnosis with hierarchical causality graph in large distributed systems. In IEEE INFOCOM 2014 (pp. 1887-1895). IEEE.
3. Chen, J., He, X., Lin, Q., Xu, Y., Zhang, H., Hao, D., et al. (2019). An empirical investigation of incident triage for online service systems. In IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (pp. 111-120). IEEE.
4. Deng, A., & Hooi, B. (2021). Graph neural network-based anomaly detection in multivariate time series. In Proceedings of the AAAI Conference on Artificial Intelligence, 35(5), 4027-4035.
5. Doshi-Velez, F., & Kim, B. (2017). Towards a rigorous science of interpretable machine learning. arXiv preprint arXiv:1702.08608.
6. Du, M., Li, F., Zheng, G., & Srikumar, V. (2017). DeepLog: Anomaly detection and diagnosis from system logs through deep learning. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (pp. 1285-1298).
7. Gama, J., Zliobaite, I., Bifet, A., Pechenizkiy, M., & Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Computing Surveys, 46(4), 44.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

8. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., Marchand, M., & Lempitsky, V. (2016). Domain-adversarial training of neural networks. Journal of Machine Learning Research, 17(59), 1-35.
9. Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., & Dahl, G. E. (2017). Neural message passing for quantum chemistry. In Proceedings of the 34th International Conference on Machine Learning (pp. 1263-1272).
10. Hamilton, W., Ying, Z., & Leskovec, J. (2017). Inductive representation learning on large graphs. Advances in Neural Information Processing Systems, 30, 1024-1034.
11. Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutierrez, C., et al. (2021). Knowledge graphs. ACM Computing Surveys, 54(4), 71.
12. Hu, Z., Dong, Y., Wang, K., & Sun, Y. (2020). Heterogeneous graph transformer. In Proceedings of The Web Conference 2020 (pp. 2704-2710).
13. Kazemi, S. M., Goel, R., Jain, K., Kobyzev, I., Sethi, A., Forsyth, P., & Poupart, P. (2020). Representation learning for dynamic graphs: A survey. Journal of Machine Learning Research, 21(70), 1-73.
14. Kipf, T. N., & Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In International Conference on Learning Representations.
15. Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In Proceedings of the IEEE International Conference on Computer Vision (pp. 2980-2988).
16. Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. Advances in Neural Information Processing Systems, 30, 4765-4774.
17. Luo, D., Cheng, W., Xu, D., Yu, W., Zong, B., Chen, H., & Zhang, X. (2020). Parameterized explainer for graph neural network. Advances in Neural Information Processing Systems, 33, 19620-19631.
18. Luo, C., Lou, J.-G., Lin, Q., Fu, Q., Ding, R., Zhang, D., & Wang, Z. (2014). Correlating events with time series for incident diagnosis. In Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1583-1592).
19. Ma, M., Xu, J., Wang, Y., Chen, P., Zhang, Z., & Wang, P. (2020). AutoMAP: Diagnose your microservice-based web applications automatically. In Proceedings of The Web Conference 2020 (pp. 246-258).
20. McMahan, B., Moore, E., Ramage, D., Hampson, S., & Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (pp. 1273-1282).
21. Pan, S. J., & Yang, Q. (2010). A survey on transfer learning. IEEE Transactions on Knowledge and Data Engineering, 22(10), 1345-1359.
22. Pearl, J. (2009). Causality: Models, reasoning, and inference (2nd ed.). Cambridge University Press.
23. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1135-1144).
24. Rossi, E., Chamberlain, B., Frasca, F., Eynard, D., Monti, F., & Bronstein, M. (2020). Temporal graph networks for deep learning on dynamic graphs. arXiv preprint arXiv:2006.10637.
25. Rudin, C. (2019). Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence, 1(5), 206-215.

BODHIVRUKSHA JOURNAL OF DIVERSE DISCIPLINE
E-ISSN : 3139-1486
An Open Access, Peer-Reviewed, Multidisciplinary Scholarly Journal
Bi-monthly | Multilingual (English, Hindi, Marathi) | Academic Research
@ Volume 2, Issue 5 (September-October 2026) (Special Issue 1) ISSN: 3139-1486 (Online)

26. Schlichtkrull, M., Kipf, T. N., Bloem, P., van den Berg, R., Titov, I., & Welling, M. (2018). Modeling relational data with graph convolutional networks. In The Semantic Web: ESWC 2018 (pp. 593-607). Springer.
27. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. In International Conference on Learning Representations.
28. Wu, L., Tordsson, J., Elmroth, E., & Kao, O. (2020). MicroRCA: Root cause localization of performance issues in microservices. In NOMS 2020 - IEEE/IFIP Network Operations and Management Symposium (pp. 1-9). IEEE.
29. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., & Yu, P. S. (2021). A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems, 32(1), 4-24.
30. Ying, Z., Bourgeois, D., You, J., Zitnik, M., & Leskovec, J. (2019). GNNExplainer: Generating explanations for graph neural networks. Advances in Neural Information Processing Systems, 32, 9240-9251.
31. Yu, G., Chen, P., Chen, H., Guan, Z., Huang, Z., Jing, L., et al. (2021). MicroRank: End-to-end latency issue localization with extended spectrum analysis in microservice environments. In Proceedings of The Web Conference 2021 (pp. 3087-3098).
32. Yu, G., Chen, P., Li, Y., Chen, H., Li, X., & Zheng, Z. (2023). Nezha: Interpretable fine-grained root causes analysis for microservices on multi-modal observability data. In Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (pp. 553-565).
33. Yuan, H., Yu, H., Gui, S., & Ji, S. (2023). Explainability in graph neural networks: A taxonomic survey. IEEE Transactions on Pattern Analysis and Machine Intelligence, 45(5), 5782-5799.
34. Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., et al. (2020). Multivariate time-series anomaly detection via graph attention network. In 2020 IEEE International Conference on Data Mining (pp. 841-850). IEEE.
35. Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2020). Graph neural networks: A review of methods and applications. AI Open, 1, 57-81.