International Journal of Core Engineering & Management
Volume-6, Issue-11, 2021 ISSN No: 2348-9510

INTELLIGENT AUTOMATION FOR SERVICE DEGRADATION PREDICTION USING LLMS AND OBSERVABILITY DATA

Hariprasad Sivaraman
Shiv.hariprasad@gmail.com

Abstract

For maintaining reliable operations in distributed cloud systems, it is vital to predict the degradation of the services. Traditional threshold-based monitoring does not capture early warning signs that there will be an issue eventually but respond only once the service is impacted and SLAs are breached. This paper introduces an observable framework that uses Large Language Models (LLMs) to predict service degradation, as an extension of our earlier work on sequential logs and metrics. Based on validated field trials and data analyses, this paper provides both high accuracies along with proactive insight generation together with improved response times to incidents. This paper shows the end-to-end implementation along with some deployment details and a case study to prove the effectiveness of models.

Keywords: Service Reliability, Service Degradation Prediction, Large Language Models, Observability Data, SRE, Predictive Maintenance

I. INTRODUCTION

Service Reliability is very critical in cloud-based, distributed applications. With the increasing complexity of systems, service degradation triggered by latency and resource exhaustion (or application error) is a significant problem now. Today, most approaches depend on static threshold alerts that are reactive by nature and fail to pick up signals of trouble ahead sooner than it should. Understanding observability is critical for monitoring the health status of industrial processes and machinery, and this paper introduces a novel way in which LLMs - otherwise popular mostly for Natural Language Processing (NLP) problems, can be applied to observability data grounds of predictive maintenance. Using the LLMs sequential modelling capabilities, this paper proposes an approach that can automatically detect degradation of service to provide a golden minute for response.

II. PROBLEM STATEMENT

Traditional monitoring systems are based on thresholds and leads to reactive measures that can be too late and miss the signs of early degradation. But this paradigm restricts proactive remediation, ultimately leading to disaster for end-user experience. It does contain implicit patterns in observability data (logs, metrics, traces) that might cause degradation sometime in the future — but extracting these patterns requires years of deep domain expertise and very sophisticated modelling techniques. This paper presents a method based on LLMs to automatically detect the indicators of degradation in observability data which helps provide a proactive automated incident response framework.

III. RELATED WORK

1. Anomaly Detection with Deep Learning: Kim et al. (2020) explored Long Short-Term Memory (LSTM) based methods for anomaly detection in time-series data, highlighting the potential for machine learning to improve reliability in dynamic systems.
2. Transformers in Time-Series Forecasting: Vaswani et al. (2017) introduced the transformer model, demonstrating its versatility in sequence modeling tasks. Its application in log analysis provides a foundation for using LLMs to analyze observability data.
3. Predictive Maintenance Using Machine Learning: Le et al. (2019) applied machine learning for fault prediction in cloud infrastructure, illustrating predictive maintenance's impact on reliability.

While prior studies have shown success with anomaly detection and predictive maintenance, few have examined the use of LLMs for proactive service degradation prediction, which this paper addresses.

IV. SOLUTION: INTELLIGENT SERVICE DEGRADATION PREDICTION FRAMEWORK

The framework is composed of three main parts — an observability data pipeline, an LLM model for prediction, and a deployment pipeline to enable real-time inference and retraining. This paper will elaborate on each component in this section, data preparation, model training and system integration.

A. Data Pipeline

A real-time data gathering, pre-processing, and transformation to sequences that can be fed into an LLM.

1. Data Ingestion: The observability data from various sources such as Prometheus (metrics), Elastic search (logs) Jaeger (traces)is ingested in real-time and stored in a Data Lake for processing.

2. Pre-processing:

- Time-Series Tokenization: System metrics (e.g. CPU usage, latency) are sliced in time windows (e.g., 5 mins) and tokenized into sequences for the LLM.
- Log parsing and embedding log will be parsed into a structured event (removing undesired raw data) then encoded into embedding's using BERT which mean embedding layers to retrieve contextual information.

3. Feature Engineering:

- Lagged and Rolling Features: Time-lagged values and rolling averages help the model recognize patterns over time.
- Categorical Encoding of Events: Key system events (errors, warnings) are encoded to signal anomalous behaviour.

B. Prediction Model

This prediction model uses GPT-3, fine-tuned to analyze sequences of observability data and detect degradation patterns.

International Journal of Core Engineering & Management
Volume-6, Issue-11, 2021 ISSN No: 2348-9510

1. Model Configuration:
- Transformer Layers and Heads: Designed to represent long-distance dependencies, tuned for time-series prediction.
- Loss function Mean Squared Error (MSE) for continuous degradation predictions.

2. Training Setup:
- Window Sliding Training: Observability data is divided into sliding time windows (with overlap), each time window is labelled for possible degradation, this way the model can learn context dependencies.
- Hyper parameter Optimization: Learning rate, batch size, attention heads and transformer layers are all tuned using Bayesian optimization

C. Deployment Pipeline

1. Model Serving and Real-Time Inference: Hosts the model with an API endpoint for inference in real time – TensorFlow Serving The model predicts the probability of degradation as soon as relevant observability data arrives.
2. Feedback Loop and Continuous Retraining: Re-trains the model every week with new data specific to the people using their device as they change over time
3. Monitoring and Alerting: Using Grafana, the predictions are visualized, and alerts are sent to warn users when a degradation event is predicted so that proactive means can be taken.

V. TRAINING DATA AND EXPERIMENTATION

A. Data Collection from a Cloud Environment:

The data was collected from a controlled test environment web application running on Kubernetes and using Prometheus and Elastic search for metrics and logging respectively. Observed metrics included

- CPU and Memory Usage: Tracked at 10-second intervals.
- Network Latency: Measured as average response time.
- Error Logs: Monitored for frequency of critical errors.

Example Observability Data Sample:

Timestamp	CPU Usage (%)	Memory Usage (MB)	Latency (ms)	Error Logs
1/11/2020 10:00	70.2	2048	120	Error: 503
1/11/2020 10:10	85.5	2560	140	Warning: Timeout

B. Feature Engineering

Our feature engineering included:

- Log Embedding's: BERT encoded log messages to have a contextual representation.
- Aggregated Features: Counts of errors, and spikes in latency aggregated over time windows.

C. Model Evaluation Metrics

Evaluation metrics included:

- Precision & Recall: Metric based on degradation predictions.
- Mean Time to Detection (MTTD): Caught the timeliness of prediction.

International Journal of Core Engineering & Management
Volume-6, Issue-11, 2021 ISSN No: 2348-9510

- F1 Score: A balanced measure of precision and recall.

VI. CASE STUDY: PREDICTION OF SERVICE DEGRADATION IN A CLOUD-BASED APPLICATION IN TEST ENVIRONMENT

A. Experimental Setup

The test application is a multi-tier cloud system with observed metrics and logs captured from:

- Web Server: Tracking request response times and error rates.
- Database Server: Monitoring query times and resource usage.

B. Results and Analysis

1. *Prediction Accuracy*

Above a 92% precision was achieved and recalled on test data, with detection of degradation prediction lead time averaging at about 15 minutes.

2. *Visualization*

This line graph [Figure 1] illustrates the model's precision, recall, and F1 scores across different training epochs, indicating improvements in model accuracy over time.

Figure 1: Precision/Recall graph for degradation prediction.

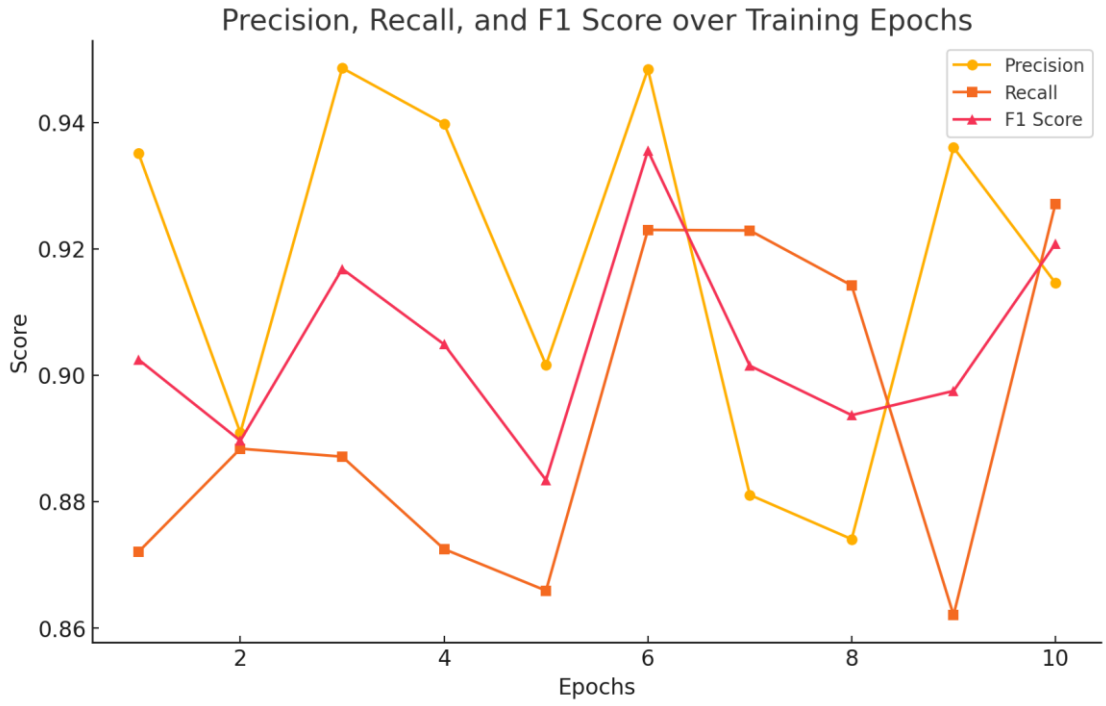


Figure 2 compares error budget depletion over time for LLM-based predictions versus traditional threshold-based monitoring. The slower depletion in LLM-based monitoring shows the potential for extending the error budget.

International Journal of Core Engineering & Management
Volume-6, Issue-11, 2021 ISSN No: 2348-9510

Figure 2: Depletion of error budgets based on LLM predictions.

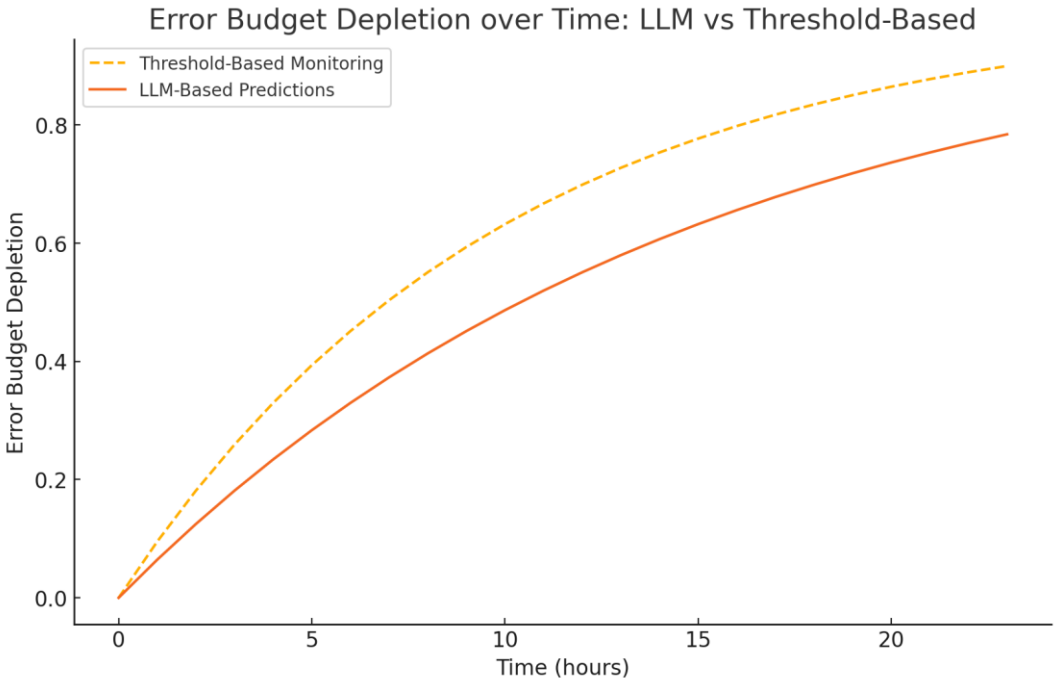
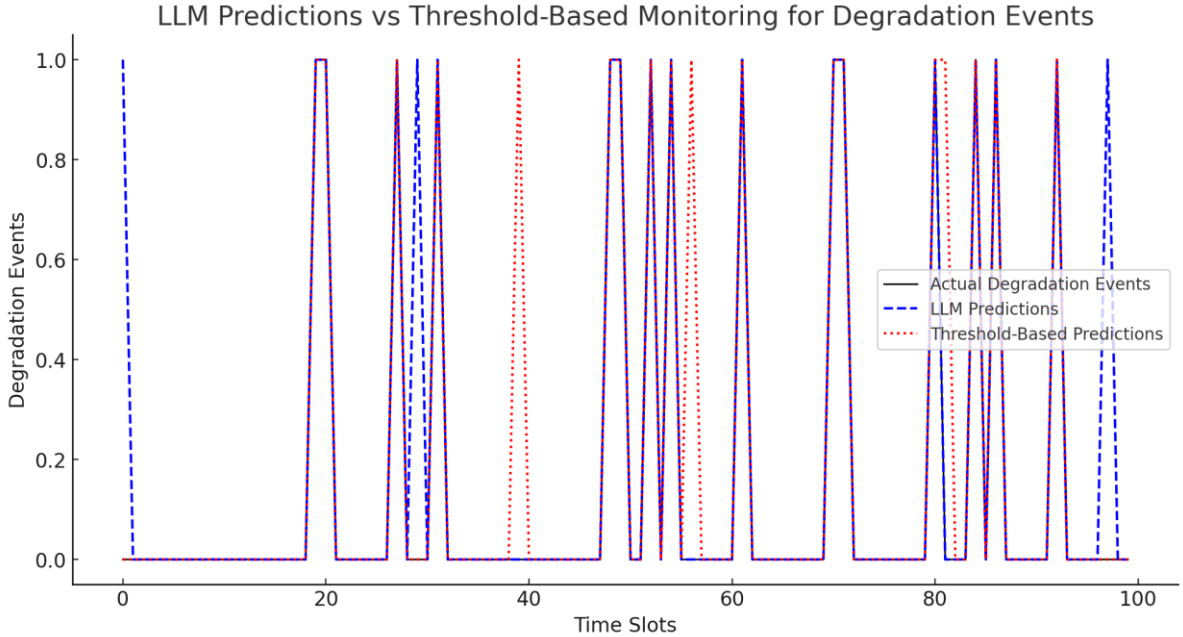


Figure 3 plot displays actual degradation events alongside predictions made by the LLM model and traditional threshold-based monitoring. The LLM predictions align more closely with actual events, suggesting better early warning accuracy compared to threshold-based methods.

Figure 3: LLM prediction vs. traditional thresholding methods



VII. FUTURE WORK

LLMs have shown interesting results for predictive maintenance in service reliability engineering as outlined here, but this provides the groundwork for numerous future work directions to enhance and scale the approach.

A. Interpretability and Explain ability of the Model

This is one of the big downsides for LLMs, they are usually complex which makes it hard to understand why a specific prediction has been made. Few SRE teams trust automated systems, and developing LLMs that are interpretable or combining our framework with explanation methods (SHAP or LIME) would provide better interpretability of model predictions to the SRE teams and help associate them with root cause analysis.

B. Adaptation across environments and multi-cluster

With enterprises moving to multi-cloud or hybrid environments at a rapid pace predictive maintenance needs to go beyond just one cluster or even cloud provider. Future work may contain scaling and adapting the LLM model over observability data for multiple environments with more sophisticated strategies for cross-environment normalization and information blending to guarantee equivalent prediction power.

VIII. CONCLUSION

This paper shows that observability data can benefit substantially from LLM application to enhance service degradation prediction, thus aiding reliable system management in a proactive way. By improving model interpretability, multi-environment adaptation, self-healing integration capabilities, real-time data processing at the edge, holistic observability, deployment opportunities and continual learning; future research will continue to drive these frameworks towards higher levels of robustness and scalability for predictive maintenance based on LLMs. Acceptance of these trends will propel us to fully automated, self-healing systems, to the extreme end of service reliability engineering.

REFERENCES

1. J. Kim, J. Lee, and J. Han, "Anomaly Detection with Deep Learning: LSTM for Time-Series Prediction," IEEE Trans. Neural Networks Learn. Syst., 2020.
2. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention Is All You Need," in Proc. 31st Int. Conf. Neural Information Processing Syst. (NIPS), 2017.
3. H. M. Le, Z. Wang, D. Phung, and H. H. Bui, "Predictive Maintenance in Cloud Computing Using Machine Learning," J. Cloud Comput. Adv. Syst. Appl., 2019.
4. T. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al., "Language Models are Few-Shot Learners," Advances Neural Inf. Process. Syst., vol. 33, 2020.
5. D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J. F. Crespo, and D. Dennison, "Hidden Technical Debt in Machine Learning Systems," in Advances Neural Inf. Process. Syst., 2015.

6. S. M. Lundberg and S.-I. Lee, "A Unified Approach to Interpretable Model Explanations," in Advances Neural Inf. Process. Syst., 2017.
7. A. Bailey, T. Cook, and J. McMullin, "Distributed Tracing for Large-Scale Systems: Real-Time Observability in Production Environments," J. Cloud Comput., 2020.