AdeptDC

Response to Project Clairvoyant RFP

Rajat Ghosh

CEO, AdeptDC, Inc

Table of Contents

I. Cover Letter *8*

II. Executive Summary *9*

Competitive Differentiation **10**

Proforma Information **10**

Company Background and Description **10**

Company's Relevant Know-How on Anomaly Detection and Failure Detection Area **11**

Company's Relevant Know-How on the Server Remediation Area **11**

III. Technical Proposal for the Solution *13*

Monitoring **15**

Failure Risk Prediction **17**

Unsupervised Anomaly Detection for Streaming Data 17
Tunable Anomaly Detection Capability 19
Multivariate Forecasting for Time series Data 19
Anomaly Detection Combined with Failure Forecasting 20
Machine Learning-Based Failure Detection on Discrete Data Streams 21
Anomaly Detection on Log Data 22
Failure Analytics on Log Data 22
Alarm Policy, Notification, and Dynamic Alarm Ranking 25
Multivariate Regression for Mean-Time-To-Failure (MTTF) Prediction 25
Multivariate Classification for Binary Failure Risk Prediction 25

Root Cause Analysis **26**

Failure Prevention Recommendation **27**

Preventive Recommendation for Online Mitigation 28
Preventive Recommendation for Offline Mitigation 29

Orchestration **30**

Scalability **30**

Algorithm 30
Software Architecture 31

IV. Project timeline from award to delivery to production ..... *Error! Bookmark not defined.*

POC Stage ..... Error! Bookmark not defined.

Testing/Integration Stage ..... Error! Bookmark not defined.

Production Deployment ..... Error! Bookmark not defined.

V. Development cost with proposed payment schedule ..... *Error! Bookmark not defined.*

POC Stage ..... Error! Bookmark not defined.

Testing/Integration Stage ..... Error! Bookmark not defined.

Production Stage ..... Error! Bookmark not defined.

VI. Hand-Off Procedure at the End of the Project – Source code, binaries, documentations 36

VII. About Us ..... *Error! Bookmark not defined.*
VIII. Appendix *38*
Machine Learning Pipeline **38**
Human-Authorized AI Action for Operational Safety **38**
Monitoring Sensor Health Scores **39**
Hierarchical Cluster Analysis **39**
Alarm Policy, Notification, and Dynamic Alarm Ranking **40**
Root Cause Analysis **41**
Impact Analysis 41
Sensor Fusion 41
Laboratory-Scale Validation of Auto-Remediation Algorithm for Online Failures **42**
Tunability **45**
AutoML Features **46**

List of Figures

Figure 1: Dashboard for AdeptDC's system health management software 13
Figure 2: AdeptDC's workflow for failure prediction and remote remediation 13
Figure 3: AdeptDC's modules and their core capabilities 14
Figure 4: AdeptDC's server monitoring window 16
Figure 5: Health score clustering analysis for a server fleet 16
Figure 6: Anomaly detection on streaming data, comprising 1252 Portworx metrics 18
Figure 7: Anomaly detection on a complex time series 18
Figure 8: Anomaly detection on multi-scale time series 18
Figure 9: Tunable anomaly detection capability 19
Figure 10: Multivariate forecasting with 2.51% RMS error for rack inlet air temperature 20
Figure 11: AdeptDC's hybrid anomaly detection combined with failure forecasting to identify early failure warning and forecast the time of failure. 21
Figure 12: Machine learning-based failure detection. The red columns indicate windows with failure risks 22
Figure 13: Failure risk detection from log data. The red columns indicate windows with failure risks. 22
Figure 14: A simulated example of term frequency analysis for log files 23
Figure 15: A simulated example for bigram frequency analysis for log files 23
Figure 16: A simulated example of latent semantic analysis for log files 24
Figure 17: A simulated example for sentiment analysis for log files 24
Figure 18: Mean-time-to-failure (MTTF) prediction module to be used as a decision support tool 25
Figure 19: Binary failure risk prediction module to be used as a decision support tool 26
Figure 20: Time series correlation ranking for a Nutanix Prism cluster 27
Figure 21: Tunable correlation ranking for a Portworx cluster. 27
Figure 22: AdeptDC's failure classification workflow 28
Figure 23: Continuous recommendation for cooling setpoint adjustments to avoid overheating 29
Figure 24: AdeptDC's NLP-based preventive recommendation engine for offline failures 30
Figure 25: Architecture diagram for the machine learning pipeline for big data algorithms 32
Figure 26: Architecture diagram for the machine learning pipeline for small data algorithms 32
Figure 27: Architecture diagram for the machine learning Pipeline for hybrid algorithms 33
Figure 28: Machine learning pipeline for the proposed predictive maintenance solution 38
Figure 29: Sensor list from a DRAC server monitored by AdeptDC 39
Figure 30: A simulated example for hierarchical cluster analysis for log files 39
Figure 31: Dynamic alarm ranking for Portworx cluster metrics 40
Figure 32: Tunable Alarm Ranking 41
Figure 33: Interactive dependency mapping between GPU utilization (independent variable) and GPU temperature (dependent variable) 41
Figure 34: Dynamic GPU temperature (^{0}C) forecasting as a function of GPU utilization (%) 42
Figure 35: Laboratory-scale test setup for the auto-remediation algorithm 43
Figure 36: Dynamic failure risk triaging for CPUs. 43
Figure 37: Monitored CPU temperature sensors for the validation case study. 44
Figure 38: Auto-Remediation sequence for CPU overheating. 44
Figure 39: Different benchmarking workloads for AdeptDC's online remediation algorithm 45

Figure 40: Tunable ranking capability 45
Figure 41: User-defined hyperparameter specifications (relative test fraction, hidden layer, batch size, epoch) for a deep neural network 46
Figure 42: User-defined hyperparameter specifications (local steps, node, batch size, epoch) for a long short term memory (LSTM) network 46

List of Tables

Table 1: Competitive differentiation for AdeptDC 10
Table 2: Module-wise break-down of AdeptDC's core capabilities 15
Table 3: Possible implementation strategies 30
Table 4: Delivery timeline (assuming May 2020 start) ..... **Error! Bookmark not defined.**
Table 5: POC cost estimate ..... **Error! Bookmark not defined.**

Glossary

AutoML: Automated parameter and model selection for machine learning.

Capacity Score: A statistical measure of the utilization level of a sensor at a particular point in time. It is expressed as (sensor value – lower control limit) / (upper control limit – lower control limit). A capacity score near 1 indicates a possible over-utilization. A capacity score near 0 indicates a possible under-utilization.

Cascading Failure: A process in a system of interconnected components where a failure in one component triggers failures in other components usually with a time delay.

Causal Map: A concept map in which nodes represent interconnected components and the links between nodes represent mutual influence

Contextual AI: A technology that can recognize human context and interact with humans

Contextual Anomaly: An unusual pattern shift for a data stream

Continuous Variable: A variable that exhibits a long-term dependency with its historical values

Corpus: A collection of log files

Decision Support Tool: A software that supports in decision-making.

Discreate Variable: A variable that does not show a long-term dependency with its historical values

Health Score: A trinity of {Capacity Score, Variability Score, Spike Score}^{i} that characterizes the health of a sensor

Log File: A file that records user-defined events that occur in a computing system

Metric: A time series that measures a system performance indicator

Offline Mitigation: A mitigation action that requires system state change by an external action such as Reboot, Reimage, Replace, Shutdown, Migration

Online Mitigation: A mitigation action that can be carried out without system downtime

Operator: A generic user persona discussed throughout this RFP response.

Risk Window: A time window for which an active failure risk exists for a component.

Runbook: A compilation of routine procedures and operations for system administrators

Sensor: A measurement point possibly both hardware and software

SMART: Self-Monitoring, Analysis and Reporting Technology

Spatial Anomaly: A violation of a health rule

Spike Score: A statistical measure of burstiness of a sensor for a time window. It is measured by (1 – Kurtosis/3) of a sensor over a time window. Numerically, a spike score is always greater than zero. The higher the value of a spike score, the higher is the component failure risk.

Streaming Data: A large number of correlated time series

Variability Score: A statistical measure of the long-term variability of a sensor for a time window. It is expressed as a ratio of standard deviation and mean. Numerically, a variability score is always greater than zero. The higher the value of a variability score, the higher is the component failure risk.

I. Cover Letter

II. Executive Summary

The end-user experience is one of the key business metrics for a leading public cloud service provider, such as [] Azure. To avoid poor customer experience from availability issues, [] solicited a proposal for a machine learning platform to predict server failures with timestamps, identify preventive actions, and deploy remote remediation. In that context, this proposal response discusses AdeptDC's pertinent capabilities, a possible engagement timeline, and a budget.

AdeptDC offers the following potential benefits for []:

- Early failure risk detection using anomaly detection and failure forecasting: AdeptDC combines unsupervised anomaly detection with failure forecasting to predict the probable timestamp for a future failure[1]. For log files, AdeptDC offers natural language processing (NLP)-based capabilities so that the operators can quickly recognize failure risks with minimal operating overhead. This capability will help [] recognize a failure warning with the corresponding time window (Figure 11) and salient topics in a corpus of log files (Figure 16).
- Multivariate failure risk prediction: AdeptDC's SaaS solution offers multiple deep learning-based multivariate failure risk prediction models such as multivariate forecasting, regression, and classification. With multivariate failure prediction models, operators can compute critical maintenance indices (e.g., MTTF and server health status) in a data-driven manner (Figure 18). AdeptDC's unique deep learning platforms will allow [] to leverage their rich labeled training data for multivariate failure model training. In addition, AdeptDC's SaaS platform comes with AutoML features to reduce analysis overhead (Figure 41).
- Dynamic root cause analysis: With AdeptDC's time series correlation ranking (Figure 20), impact analysis, and failure domain analysis, [] can recognize the potential causal factors behind a failure risk. Unlike traditional monitoring and observability providers, AdeptDC can capture cascading failure event flow so that the operators can perform a system-wide root cause analysis.
- Preventive failure remediation: AdeptDC's deep learning-based preventive recommendation engine recognizes suitable offline remediation strategies (Figure 24) for an existing failure or a potential future failure. For online failure remediation, AdeptDC offers a reinforcement learning-based service (Figure 23). With AdeptDC's preventive recommendation engine, [] can remediate failure risks faster with minimal operating overhead.
- Remote remediation: AdeptDC integrates seamlessly with several orchestration platforms such as virtual machine management systems, cloud computing platforms, network management systems, and building management systems to deploy remote remediation.

[1] An anomaly from an unusual motherboard temperature spike serves as an early warning to PROCHOT thermal throttling.

Competitive Differentiation

There are different competitors on the market. Table 1 summarizes AdeptDC's differentiators with respect to its competitors.

Table 1: Competitive differentiation for AdeptDC

Feature	AdeptDC	Competitor	AdeptDC's Benefit
Failure Prediction	Contextual failure risk detection[2]	After-the-fact failure detection	Early risk detection and zero overhead for health rules
	Unified prediction model	Prediction model requiring local parameter tuning	Scalable analytics for heterogenous data[3]
Root Cause Analysis	Shape-based similarity detection across time series[4]	Pointwise correlation computation across time series	Ability to capture cascading failure event flow
Failure Remediation	AI-based preventive recommendation based on how similar tickets have been closed in the past	Historical analysis and runbook scripts	Lower manual overhead

Proforma Information

Company Name	AdeptDC, Inc
Company Contact Person Name	Rajat Ghosh
Company Contact Person E-Mail	rajat.ghosh@adeptdc.com
Company Contact Person Phone Number	404.697.5789
Company Contact Person Title	Co-Founder/CEO

Company Background and Description

AdeptDC helps infrastructure engineers avoid unplanned downtime in large-scale distributed systems. Our contextual AI software can parse logs and telemetry data intelligently to predict future failures and recommend mitigation strategies for remote remediation. It is vendor-agnostic and works for different hardware and software platforms as well as workload types. By combining streaming and batch analyses, it offers a unified analytics stack.

[2] A contextual failure risk is an early warning for a failure. For example, a sudden rise in motherboard temperature is an early warning to a thermal trip.
[3] Hardware data from a distributed infrastructure show different variation patterns like intermittent, flaky, continuous, etc.
[4] In a distributed infrastructure, different metrics move at different speeds. Our shape-based similarity ranking can capture similarities between two time series even if they are moving at different speeds.

AdeptDC is built upon the research out of a Georgia Tech PhD which developed an unsupervised machine learning algorithm for high-dimensional monitoring and distributed resource optimization for heterogenous data center infrastructure, including both hardware and software stacks. Using a National Science Foundation Small Business Innovation Research (NSF-SBIR) grant, we brought the algorithm to production. Our academic roots are complemented by our work experience in big tech companies such as Google, [], Facebook, Amazon, IBM, and Internap.

We have successfully worked with public companies such as Arrow Electronics, Vertiv, and Nutanix. Our AI software is proven to be effective for various workload types. Our success at Arrow Electronics[5] drew media attention[6,7] .

Company's Relevant Know-How on Anomaly Detection and Failure Detection Area

- PhD in machine learning applications for data center reliability management
- National Science Foundation Small Business Innovation Research (NSF-SBIR) Award
- 10+ years of industry experience in companies like Facebook, [], Amazon, Google, Internap
- Successful deployment experience in three public companies
- 11 publications[8,9,10,11] and a pertinent US patent[12,13]

Company's Relevant Know-How on the Server Remediation Area

From its commercial deployment success, AdeptDC has gained hands-on experience with a wide range of servers and related infrastructure systems[14], as follows:

- IT hardware:
    - *HPE ProLiant servers*: DL980 G7, DL580 G8, DL380 G8, DL380 G7, DL380-G5

[5] https://www.youtube.com/watch?v=cb4DjGG2qcE&t=1s
[6] https://www.datacenterknowledge.com/machine-learning/adeptdc-wants-use-machine-learning-prevent-data-center-outages
[7] https://www.datacenterknowledge.com/machine-learning/smart-assistant-data-center-cooling
[8] https://scholar.google.com/citations?user=n5SEuIYAAAAJ&hl=en
[9] Rajat Ghosh, Y. J. (2014). "Proper orthogonal decomposition-based modeling framework for improving spatial resolution of measured temperature data." IEEE Transactions on Components, Packaging and Manufacturing Technology **4**(5): 848-858.
[10] Rajat Ghosh, Y. J. (2014). "Rapid temperature predictions in data centers using multi-parameter proper orthogonal decomposition." Numerical Heat Transfer, Part A: Applications **66**(1): Numerical Heat Transfer, Part A: Applications.
[11] Rajat Ghosh, Y. J. (2012). "Error estimation in POD-based dynamic reduced-order thermal modeling of data centers." International Journal of Heat and Mass Transfer **57**(2): 698-707.
[12] https://patents.google.com/patent/US10439912B2/en
[13] Rajat Ghosh, Y. K. J. (2019). Systems and methods for intelligent controls for optimal resource allocation for data center operations. US Patent.
[14] On average, AdeptDC takes a week for data ingestion from a new hardware or software platform.

    - *Other HP/HPE brands*: HP ProCurve, HPE StoreOnce Storage, HP Blade Server, HPE Moonshot
    - *IBM xSeries servers*: x3750, x3650, x3550, x3250
    - *Other IBM brands*: Blade servers, FlashSystem 840, FlashSystem 900
    - *Cisco Nexus switches*: 3000 series, 5000 series
    - *Brocade*: Brocade Fibre DCX, Brocade FastIron
    - *Other brands*: Lenovo ThinkServer RD550, Dell PowerEdge, Supermicro
- Virtualization platforms: VMware vSphere
- Cloud computing platforms: Amazon AWS, [] Azure, Digital Ocean
- Orchestration platforms: Kubernetes, Portworx, Nutanix Prism
- Power/Cooling systems: Daikin, Stulz, Liebert, Eaton, APC
- Control systems: Honeywell, Tridium, Automated Logic
- Communication interfaces/protocols: IPMI, SNMP, SSH, BACnet, Modbus

III. Technical Proposal for the Solution

Hardware error caused system performance impact is common problem for a large-scale infrastructure. AdeptDC offers an AI software for end-to-end failure management starting from failure risk detection to remote remediation. Figure 1 shows a screenshot of the dashboard from AdeptDC's solution. It includes Failure Landscape which reports the number of incidents over time; Incident Mapping which tabulates the incident origination silos (e.g., Application Layer, Communication Layer, Virtual Layer, or Physical Layer); and QoS Overview including summaries of Spike, Variability, and Utilization Status.

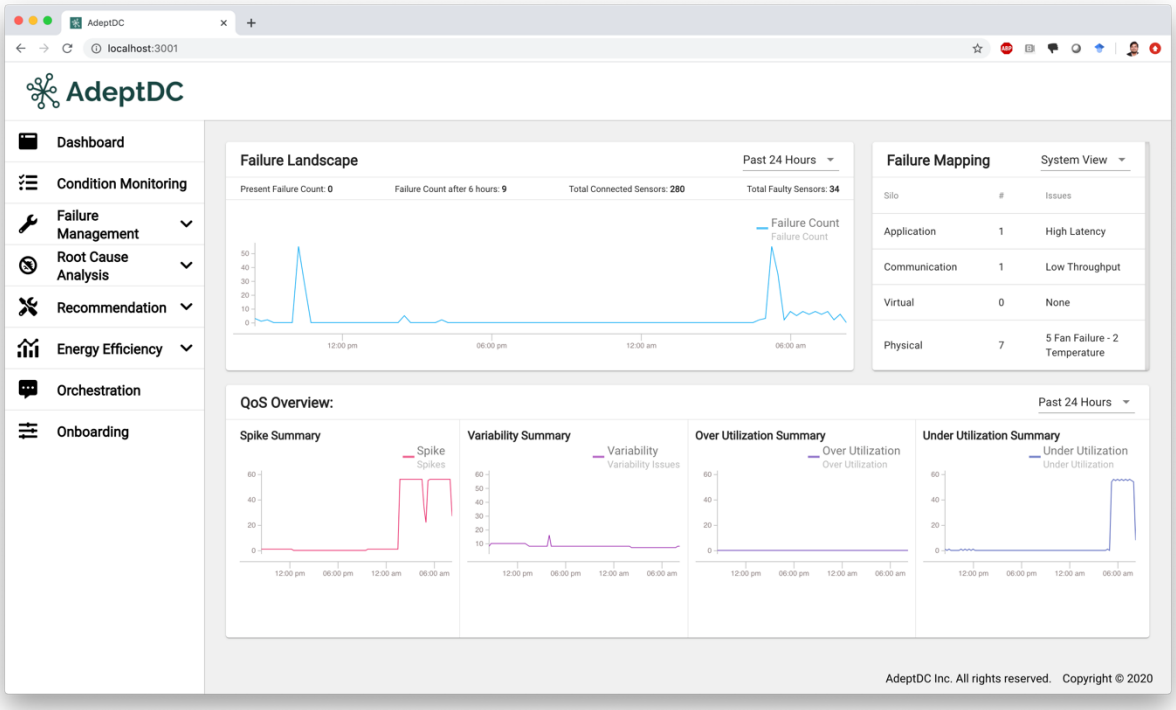


Figure 1: Dashboard for AdeptDC's system health management software

AdeptDC offers an end-to-end workflow for failure prediction and remote remediation. The workflow is enabled by five modules: Monitoring, Failure Risk Forecasting, Root Cause Analysis, Recommendation Engine, and Orchestration, as shown in Figure 2.

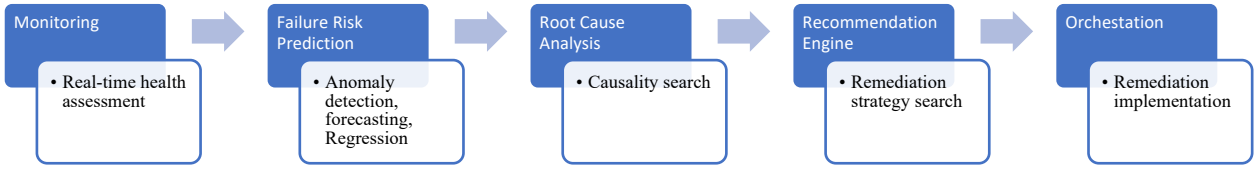


Figure 2: AdeptDC's workflow for failure prediction and remote remediation

- Monitoring: Monitoring informs real-time system status. For rapid health assessment of different service components with large degrees of freedom, this module converts a monitored time series data into standardized health scores, namely capacity score, variability score, and spike score.
- Failure Risk Prediction: Failure risk prediction acts by analyzing the probability of a future failure for both continuous and discrete variables.
- Root Cause Analysis: Root cause analysis identifies the causal factors for a predicted failure risk.
- Recommendation Engine: The recommendation engine uses machine learning to identify a suitable mitigation strategy for a failure risk.
- Orchestration: The orchestration layer implements remote remediation actions computed by the recommendation engine.

The five modules, as shown in Table 2 and Figure 2, are discussed in detail in the following sections.

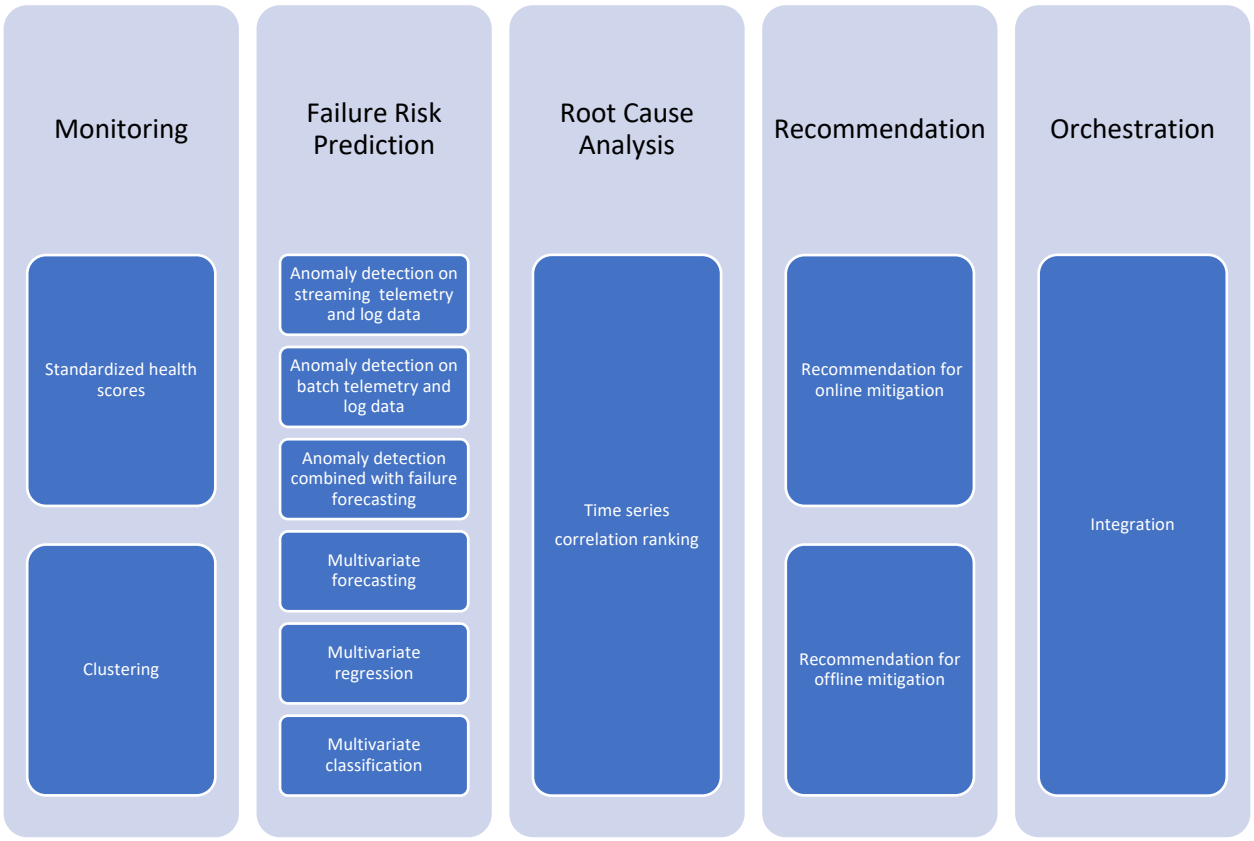


Figure 3: AdeptDC's modules and their core capabilities

Table 2: Module-wise break-down of AdeptDC's core capabilities

Module	Sub-Module	Purpose
Monitoring	Standardized health score	Unified health risk assessment
	Clustering	System-wise health risk analysis
Failure Risk Prediction	Anomaly detection on streaming telemetry and log data	Early risk identification without health rules
	Anomaly detection on batch telemetry and log data	Unusual pattern detection
	Anomaly detection combined with failure forecasting	Early risk detection with risk window
	Multivariate forecasting	Concurrent future risk assessment
	Multivariate regression	Parametric failure risk prediction
	Multivariate classification	Parametric failure risk classification
Root Cause Analysis	Time series correlation Ranking	Causality recognition
Recommendation Engine	Preventive recommendation for online mitigation	Automated online remediation identification
	Preventive recommendation for offline mitigation	Automated offline remediation identification
Orchestration	Integration	Remote remediation implementation

Monitoring

Telemetry and log files (e.g., system event logs) come with a wealth of data. But they hardly communicate any contextual information such as health scores, making it hard to make cross-sensor comparisons. For that purpose, AdeptDC converts monitored data into standardized health scores. Figure 4 shows AdeptDC's server health monitoring window, including device information together with user-customizable tags, incident status, workflow navigation, histogram for server health, clustering of different sensor metrics, sensor list, standardized health scores, and a collaboration tool. AdeptDC computes three standardized health scores such as Capacity Score, Variability Score, and Spike Score.

- Capacity Score: The Capacity Score estimates a real-time statistical measure of the utilization level of a sensor. It is defined as the ratio of the difference between the real-time sensor value and the lower control limit, and the difference between the upper and lower control limits. A capacity score near 1 indicates a possible over-utilization. A capacity score near 0 means a possible under-utilization.
- Variability Score: The Variability Score estimates the variability of a sensor over a time window. It is measured by the ratio of the standard deviation and the mean for a sensor. Numerically, a Variability Score is always greater than zero. The higher the value of a variability score, the higher is the component failure risks.

- Spike Score: The Spike Score estimates the burstiness of a sensor over a time window. It is measured by (1 – Kurtosis/3) of a sensor over a time window. The higher the value of a Spike Score, the higher is the component failure risk.

The trinity of {Capacity Score, Variability Score, Spike Score} defines the health of a sensor. Figure 29 in the Appendix shows the sensor health score list from a DRAC server.

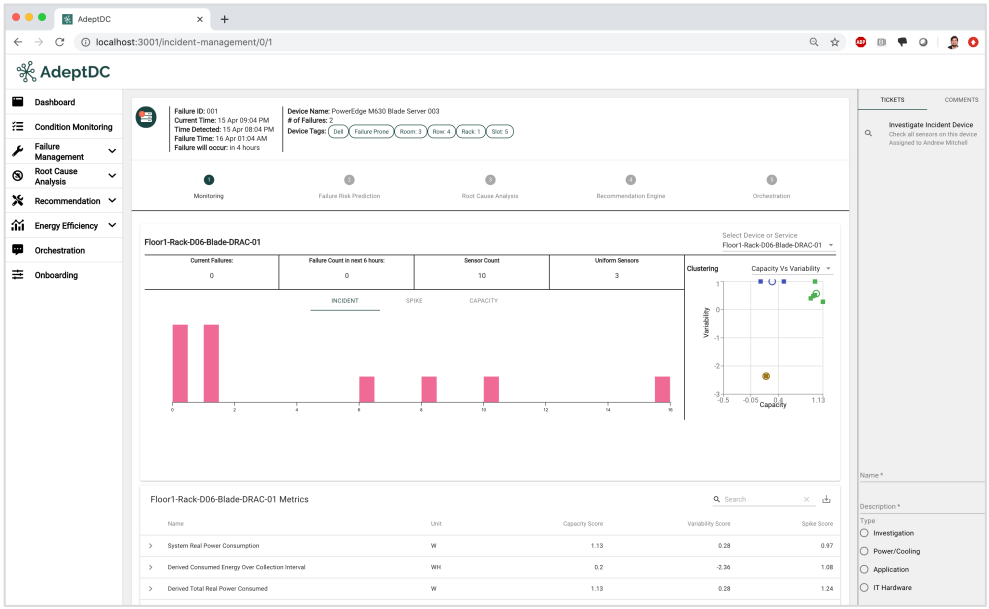
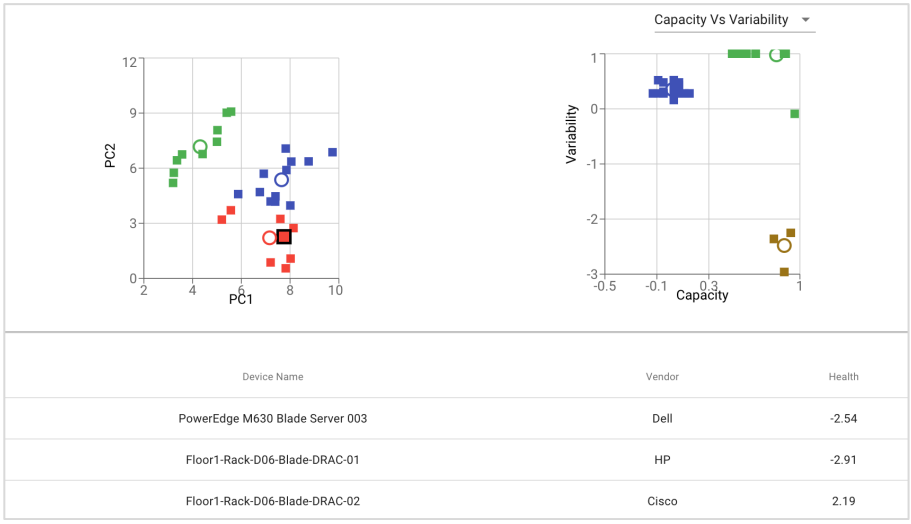


Figure 4: AdeptDC's server monitoring window



Device Name	Vendor	Health
PowerEdge M630 Blade Server 003	Dell	-2.54
Floor1-Rack-D06-Blade-DRAC-01	HP	-2.91
Floor1-Rack-D06-Blade-DRAC-02	Cisco	2.19

Figure 5: Health score clustering analysis for a server fleet

AdeptDC offers a clustering platform to recognize the latent inter-dependencies among {Capacity Score, Variability Score, Spike Score} across different clusters. Figure 5 shows the

screenshots from AdeptDC's clustering platform. The clustering platform also includes unified health scores[15] for servers.

Failure Risk Prediction

The availability and service quality of a public cloud platform like [] Azure depend on the underlying infrastructure health. In this context, a server failure risk prediction capability can reduce expensive downtime risks and minimize maintenance overhead. AdeptDC offers a comprehensive failure risk prediction capability for both telemetry and log data. It works for both real-time streaming data and batch data.

Unsupervised Anomaly Detection for Streaming Data

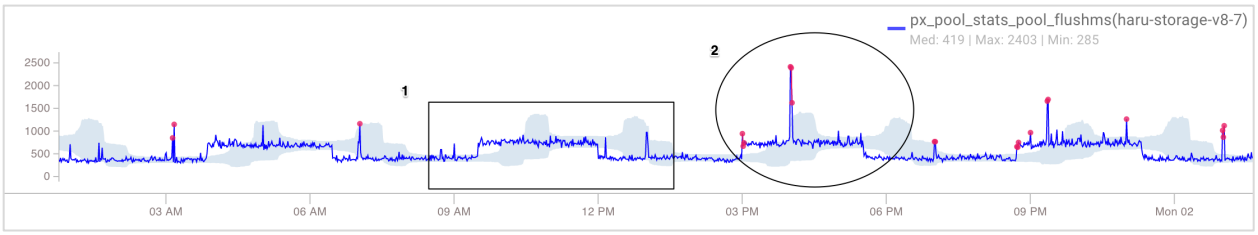
Unsupervised anomaly detection deals with the detection of an unusual pattern without a health rule. An anomaly serves as an early warning of a failure[16]. In addition, a large-scale distributed computing environment like [] Azure infrastructure can be characterized by streaming data which essentially is a large number of inter-related time series data streams, continuously generated by different sources. It is challenging for an anomaly detection algorithm to handle the intrinsic variety of streaming data within a reasonable latency constraint. Figure 6 shows AdeptDC's high-accuracy anomaly detection for streaming data, comprising 1252 Portworx metrics. Figure 6, which is a screenshot from AdeptDC's anomaly detection module, shows anomaly detection results for a write-throughput metric and a num-flushes metrics. Although they show distinctively different variation patterns, AdeptDC can detect anomalies in both of them with minimal false positive rates. This demonstrates the wide applicability of AdeptDC's anomaly detection capability.

A typical anomaly detection tool encumbers its users with high false positive rates from complex seasonal patterns and irregular spikes. As shown in Figure 7 (a screenshot from AdeptDC's anomaly detection module), AdeptDC can handle both of these challenges. It shows the anomaly detection result for Portworx[17] flush latency for a 24-hr. time range. Sub-window 1 shows a complex seasonal pattern with zero false positives. Sub-Window 2 shows irregular spikes where the red dots indicate contextual anomalies[18].

Another phenomenon that confuses typical anomaly detection tools is multi-scale time series data. These multi-scale time series are common for traffic metrics such as disk throughput. AdeptDC's anomaly detection capability can easily handle multi-scale time series as shown in Figure 8 which is a screenshot of a 24-hr. anomaly detection result for a Portworx flushed-bytes metric. As shown in the sub-window, the metrics vary from 20M to 2.0G.

[15] A unified health score combines Capacity Score, Variability Score, and Spike Score, as discussed in the Monitoring section.
[16] An unusual motherboard temperature spike serves as an early warning for PROCHOT thermal throttling.
[17] A storage environment in Kubernetes ecosystem. https://portworx.com/
[18] A contextual anomaly, as opposed to a spatial anomaly or rule violation, indicates an unusual pattern.

Figure 6: Anomaly detection on streaming data, comprising 1252 Portworx metrics



1. Complex seasonality
2. Irregular spikes

Figure 7: Anomaly detection on a complex time series

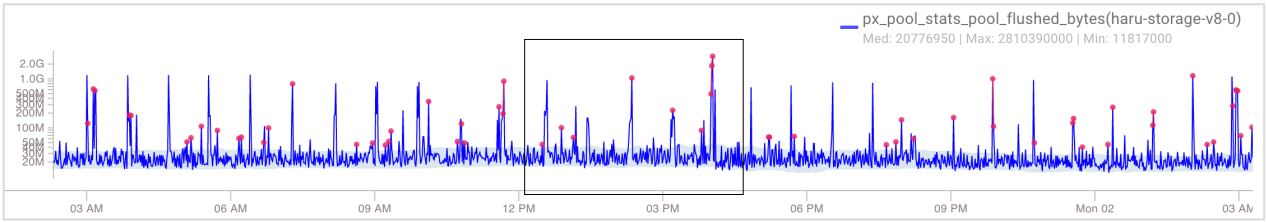
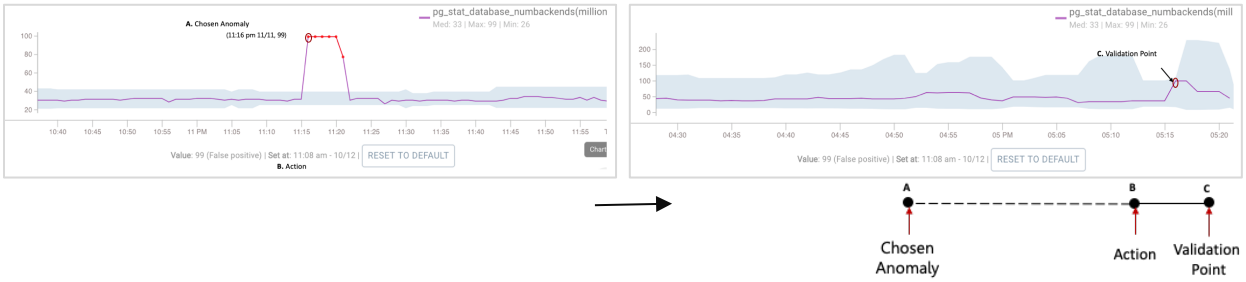


Figure 8: Anomaly detection on multi-scale time series

The blue bands in our anomaly detection results (Figure 6, Figure 7, Figure 8) indicate dynamic baselines computed by AdeptDC. Any value outside the band is an anomaly candidate. There are quite a few points outside the blue bands not marked as anomalies by red dots. These points match either a periodic pattern, as uncovered by AdeptDC, or a user-defined pattern.

Tunable Anomaly Detection Capability

A typical anomaly detection tool produces a lot of false positives. To eliminate these false positives, a user has to deal with burdensome hyperparameter tuning. To help users with hyperparameter tuning, AdeptDC offers a declarative feedback transfer capability which can accept feedback at the UI layer and automate low-level parameter tuning. Figure 9 demonstrates AdeptDC's feedback transfer capability in action; a user can mark a past anomaly as a false positive. Based on that feedback, our AI engine will automatically adjust the baseline and minimize future false positive rates.



A. A past anomaly point of value 99, marked as a false positive.
B. The time when the user took the feedback action.
C. A validation point (no anomalies for the chosen value of 99).

Figure 9: Tunable anomaly detection capability

Multivariate Forecasting for Time series Data

In a distributed computing environment like [] Azure, multiple transient variables vary together and form so-called streaming data as we have discussed in the context of anomaly detection. Just like anomaly detection for streaming data, AdeptDC offers multivariate forecasting. It reduces the operating burden by forecasting multiple variables together and analyzes their failure risk in an inter-dependent manner. Figure 10 shows the validation result. Figure 10(a) depicts a rack[19] inlet air temperature distribution with temperature sensors represented by black circles. Figure 10(b) shows the corresponding temperature forecasting by AdeptDC's machine learning model (POD[20]). Figure 10(c) reflects the forecasting error distribution with a percentage root mean square error (PRMSE) of 2.51%. The low PRMSE indicates a high accuracy of the forecasting model.

[19] A rack represents 42 1U servers.
[20] POD is one of our forecasting models.

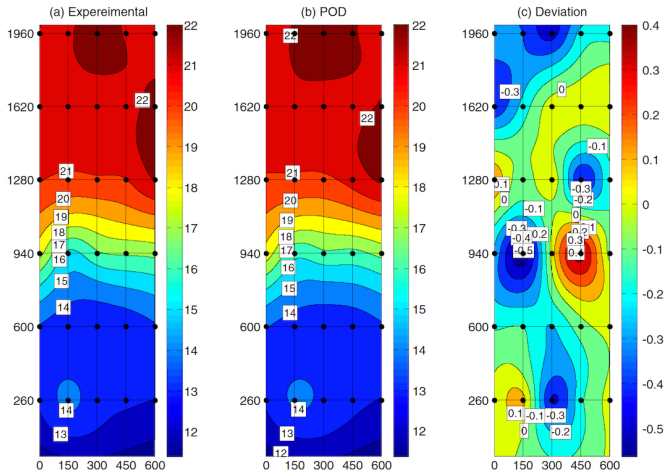


Figure 10: Multivariate forecasting with 2.51% RMS error for rack inlet temperature

Anomaly Detection Combined with Failure Forecasting

Although a typical anomaly detection tool sends an early warning for a future failure, it is not particularly actionable without a failure forecast which tells the operators when the failure is going to happen. Such a forecast allows the operators to recognize a risk window and determine the urgency of a failure warning. AdeptDC combines anomaly detection with failure forecasting[21] to offer such a risk window identification capability. Figure 11 shows a screenshot from AdeptDC's anomaly detection combined with our failure forecasting capability. It shows GPU and CPU temperature anomalies (transitions from green to yellow bars) and corresponding risk windows (yellow bars). AdeptDC's contextual AI is aware that both the GPU and CPU belong to the same server box, and, therefore, models their temperatures in a highly correlated manner. The early failure warnings, in the form of anomalies, were detected in the GPU at 8:47 AM and in the CPU at 9:17 AM. It was also forecasted that the GPU temperature would violate its reliability threshold of 95 ^{0}C at 2:32 PM and the CPU temperature would cross its reliability threshold of 75 ^{0}C at 3:02 PM. This means the risk windows (yellow bars) for both the GPU and CPU units are equal to 5 hr. 45 min. The solid blue lines in the GPU and CPU temperature charts represent telemetry data. The dotted lines in the GPU and CPU temperature charts represent the forecasted time series. This capability also validates AdeptDC's effectiveness in predictive cascading failure management (the example in Figure 11 represents a thermal runaway[22] event). With this predictive capability, the operators can quickly determine the most effective mitigation strategy.

[21] For forecasting, AdeptDC uses an ensemble approach, comprising LSTM, Holt-Winter.
[22] Increase in temperature triggers more temperature rise.

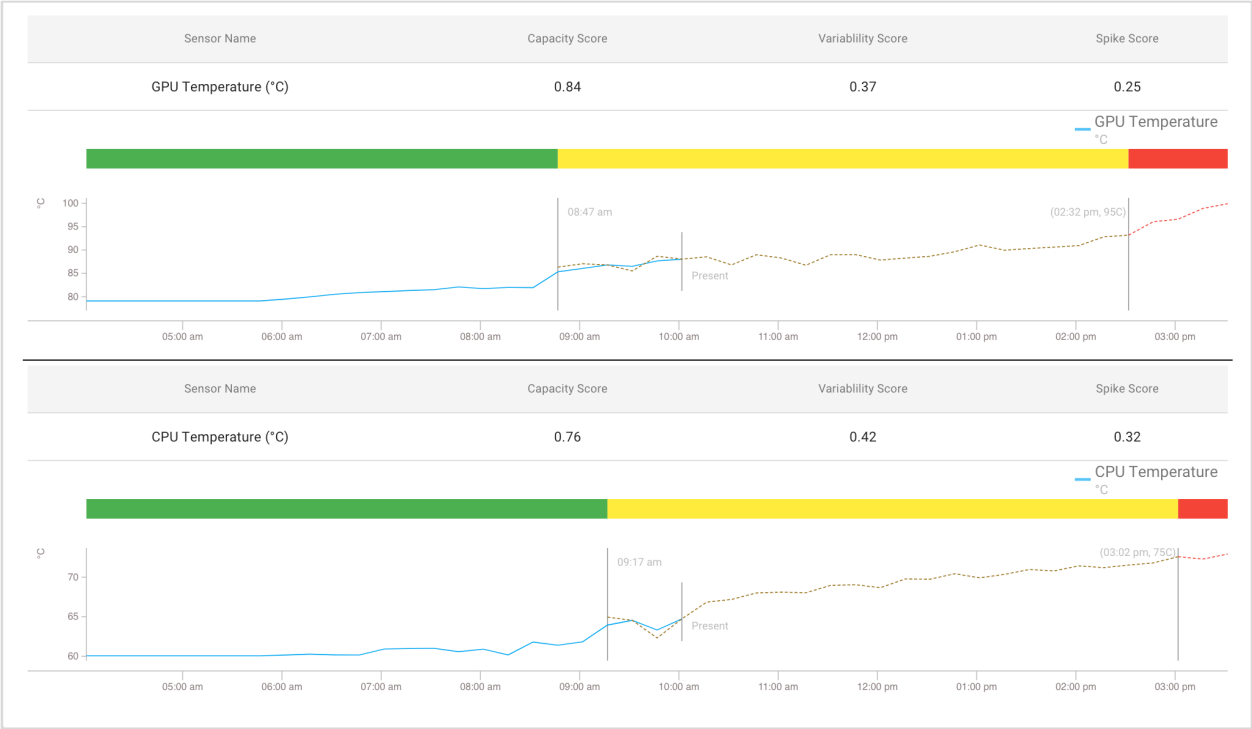


Figure 11: AdeptDC's hybrid anomaly detection combined with failure forecasting to identify early failure warning and forecast the time of failure.

There are three vertical lines in the two charts representing different time points. The first line indicates the early warning, the second line indicates the present time (when this snapshot was taken), and finally, the last line indicates the forecasted time of violation.

Machine Learning-Based Failure Detection on Discrete Data Streams

A discrete data stream does not show a long-term temporal dependency, unlike a continuous data stream. A typical example of a discrete data stream is an error count. Because a discrete data stream does not show a long-term dependency, a contextual anomaly detection analysis offers little predictive value. That said, failure risk detection for a discrete data stream is useful for recognizing recurring error patterns. For that purpose, AdeptDC offers a machine learning/deep learning[23]-based capability which learns from historical failures and classifies whether a new error might cause a failure. Figure 12 shows AdeptDC's failure detection capability for HTTP error (4 Xx request). It counts the number of errors over an hour. Although there were non-zero error counts between 7-8 AM and 9-10 AM, the failures are marked for intervals between 11 AM-12 PM, 12-1 PM, and 1-2 PM. Essentially, a non-zero error count does not amount to a failure. In fact, often an isolated error comes from component noise and system randomness. Our DNN can differentiate between true failures and isolated errors.

[23] If more than 5000 labeled data are available, we would prefer deep learning over machine learning (vanilla logistic regression).

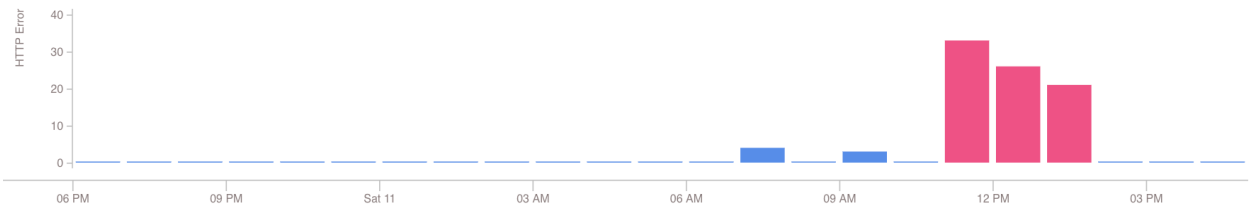


Figure 12: Machine learning-based failure detection. The red columns indicate windows with failure risks.

Anomaly Detection on Log Data

A failure event is often accompanied with a large volume of logs from a variety of sources. However, the volume and obscurity of log data make them hard for humans to comprehend easily. Typically, there are no explicit early warning messages in logs. A human operator has to recognize unusual patterns for early risk detection. For example, a failure event could be a FPGA Self-Test Failed. A corresponding early warning might be FPGA too hot (critical temperature reached). AdeptDC's contextual AI software can recognize unusual text patterns from logs. It computes keyword[24] frequencies, normalized by relevance[25], and compares them between adjacent time windows. Depending on the availability of labeled training data, a detected unusual pattern is passed through a deep neural net (DNN) or a logistic regression unit to determine if it represents a true failure or not. Figure 13 shows a representative result from AdeptDC's failure risk detection module for log files based on keyword frequency patterns and DNN. The red columns, in Figure 13, represent time windows with anomalies. It shows although there are non-zero keyword counts in several time window (7-8 AM, 9-10 AM, 12-1 PM, 2-5 PM), the anomalies are coming from log files at 7-9 PM.

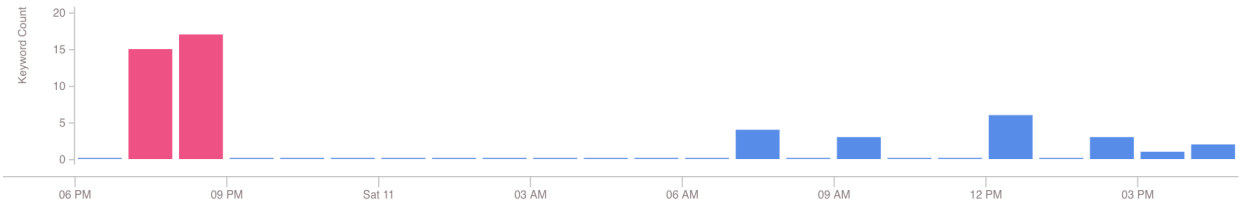


Figure 13: Failure risk detection from log data. The red columns indicate windows with failure risks.

Failure Analytics on Log Data

Due to volume, variety, and unstructured format, log data presents challenges for human comprehension. AdeptDC offers comprehensive log analytics capabilities, including: name entity

[24] A corpus of keywords will be put together based on the supplied error code.
[25] The probability of occurrence in a certain time window.

recognition/term frequency analysis, latent semantic analysis, sentiment analysis, and hierarchical clustering analysis

Named Entity Recognition/Term Frequency Analysis

Due to volume, variety, and unstructured format, it is challenging to recognize keywords from log files. To address this challenge, AdeptDC offers a named entity recognition capability which predicts keywords such as component names, error types, etc. Then, AdeptDC rapidly ranks the keywords rapidly based on their counts. Figure 14 shows a representative diagram from AdeptDC's named entity recognition/term frequency analysis capability. It shows the top-ranked keywords are {ssd, fpga, smart} of types {component, component, error}. The means potential high failure risks in SSD and FPGA. An isolated single keyword is useful, but a contiguous group of words (an n-gram) coveys more contextual information for complex log files. Similar to a single keyword analysis capability as shown in Figure 14, AdeptDC features bigram (as shown in Figure 15) and n-gram analysis capabilities. It shows top-ranked bigrams as {ssd internal, fpga hot}. This means that potential high failure risks are coming from internal ssd and hot fpga. As can be seen, bigram expresses more contextual information about failures than an isolated single keyword.

Rank	Keyword	Type
1	ssd	component
2	fpga	component
3	smart	error
4	disk	component
5	temperature	error
6	ecc	error
7	bios	component
8	diskpart	component
9	motherboard	component
10	bmc	component

Figure 14: A simulated example of term frequency analysis for log files

Rank	Bigram	Type
1	ssd internal	component
2	fpga hot	error
3	smart temperature	error
4	disk capacity	error
5	controller temperature	error
6	ecc read	error
7	bios fw	component
8	diskpart media	component
9	motherboard thermal	error
10	bmc fw	component

Figure 15: A simulated example for bigram frequency analysis for log files

Latent Semantic Analysis

While keywords or contiguous keywords are important for failure predictions, they are often highly correlated semantically and lead to significant analysis overhead. AdeptDC offers a latent semantic analysis capability which helps operators identify core issues (aka topics, in NLP verbiage) from a large corpus of log files. Figure 16 shows a representative result from AdeptDC's latent semantic analysis capability. It shows that the top three topics involve words such as {battery, smart, motherboard}. The words forming a topic are weighted to show their salience. In the first topic, "battery" is the most salient word. From this latent semantic analysis, an operator could quickly conclude that the top three failure risks are coming in relation to battery power, disk capacity, and motherboard thermal.

Topic Rank	Words
1	-0.25*battery - 0.21*cmos - 0.17*rtc - 0.12*power - 0.11*interrupt
2	-0.27*smart - 0.22*disk - 0.19*capacity - 0.14*ecc - 0.09*nand
3	- 0.35*motherboard - 0.32*thermal - 0.18*trip - 0.11*fault - 0.04*investigation

Figure 16: A simulated example of latent semantic analysis for log files[26]

Sentiment Analysis

Operators often have to spend a long time in assessing if a particular log message, a keyword, a phrase, or a topic requires their attention. To relieve this operating burden, AdeptDC offers a sentiment analysis capability which classifies a log message into two risk groups: POSITIVE (RISKY) and NEGATIVE (NOT RISKY). Figure 17 shows a representative diagram from AdeptDC's sentiment analysis capability. It shows two failure messages have been correctly assigned with a POSITIVE sentiment while a successful HTTP request has been correctly marked with a NEGATIVE sentiment.

Log Message	Sentiment
Thermal sensor on MB exceeded threshold	POSITIVE
GET / HTTP/1.1 200	NEGATIVE
NVME DMA failure	POSITIVE

Figure 17: A simulated example for sentiment analysis for log files

Hierarchical Cluster Analysis

AdeptDC offers hierarchical clustering capabilities so that a user can easily drill down a corpus[27] of log files. AdeptDC's hierarchical clustering capability is discussed in detail in the Appendix (Figure 30).

[26] The numerical values for the scalar weights are purely for illustrative purposes.
[27] A collection of log files, common in NLP verbiage

Alarm Policy, Notification, and Dynamic Alarm Ranking

Based on the anomalies, AdeptDC produces alarms. AdeptDC's alarm management capability is described in the Appendix (Figure 31, Figure 32).

Multivariate Regression for Mean-Time-To-Failure (MTTF) Prediction

Mean-time-to-failure (MTTF) for a server is a critical maintenance index for an infrastructure. Heuristic-based MTTF computations often produce inaccurate results because MTTF often depends upon multiple variables with complex correlations. To avoid heuristics, AdeptDC offers a deep neural network (DNN)-based regression capability[28] which predicts multivariate indices such as MTTF with high accuracy. Figure 18 shows a screenshot from AdeptDC's DNN-based regression capability. It shows MTTF (in month) being predicted as a parametric function of Service Age (month), Protection Patch[29], Capacity Score, Variability Score, and Spike Score, as shown in Figure 18. With this model, an operator can assess how different parameters impact an output variable such as MTTF in data-driven manner.

With this DNN-based capability, companies can leverage the rich labelled data at their disposal. This is AdeptDC's unique capability in the application performance market. AdeptDC's platform trains the model with minimal analysis overhead for its users. AdeptDC's also offers model tuning capability so that users can update the regression models based on their specific requirements (Figure 41).

Results:

Age	Protection Patch	Capacity Score	Variability Score	Spike Score	MTTF (month)
28	0	0	0.34	0.23	0.62
26	0	0	0.22	0.27	1.79
30	1	0.7	0.52	0.58	27.75
26	1	0.72	0.64	0.25	28.26
20	1	0.12	0.67	0.17	4.44
24	1	0.45	0.55	0.35	**0.99**

Figure 18: Mean-time-to-failure (MTTF) prediction module to be used as a decision support tool

Multivariate Classification for Binary Failure Risk Prediction

Operators often have to decide whether a server will survive in a new operating condition. Given the complex correlations among operating variables and a failure state, operators often have to rely on error-prone heuristics. To avoid heuristics, AdeptDC offers a deep neural network (DNN)-based classification capability which predicts multivariate indices such as server failure state with high accuracy. Figure 19 shows a screenshot from AdeptDC's DNN-based classification capability. It shows server failure state being predicted as a function of different

[28] AdeptDC offers a fully managed software-as-a-service capability with minimal operating overhead for users.
[29] A protection patch is represented by a binary variable. An example of protection patch could be whether a security software is installed in a server or not.

parameters such as Service Age (month), Capacity Score, Variability Score, and Spike Score, as shown in Figure 19. With this model, an operator can assess how different parameters impact an output variable such as server failure state in data-driven manner. Just like regression, AdeptDC's classification models can be easily tuned by its users based on their specific requirements (Figure 41).

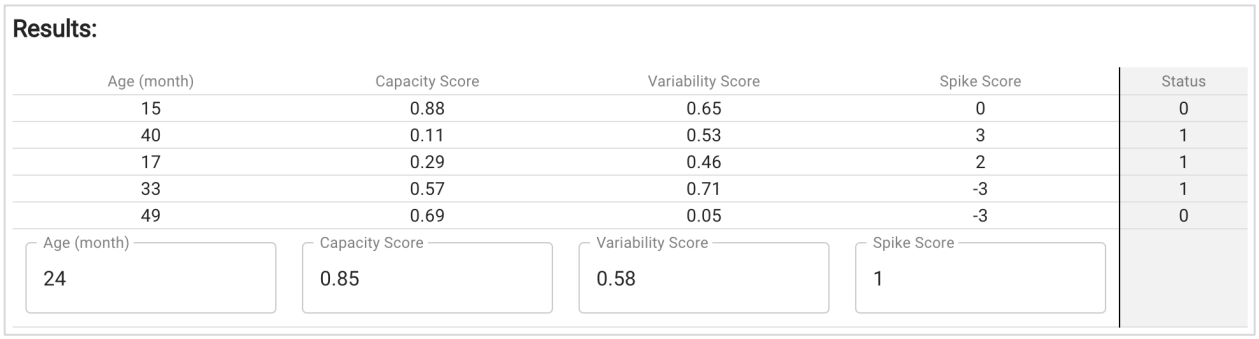


Figure 19: Binary failure risk prediction module to be used as a decision support tool

Root Cause Analysis

AdeptDC's root cause analysis capability offers a time series correlation ranking algorithm which first computes shape-based correlations across different metrics and ranks them based on the computed correlation distance. The larger the correlation distance the lower the rank. Figure 20 shows time series correlation ranking for a Nutanix Prism cluster. It ranks the correlations between 'controller_avg_write_io_latency_usecs' and all 1328 metrics in the cluster. Unlike other vendors, AdeptDC's time series correlation ranking is based upon a dynamic time warping-based similarity ranking algorithm which can handle time-lag and scale variance. As shown in Figure 20, the lengths of 'controller_avg_write_io_latency_usecs' (5:02AM-5:50AM) and 'controller_avg_write_io_latency_usecs_vdisk…' (5AM-5:58AM) time series are different. The amplitude scales for 'controller_avg_write_io_latency_usecs'(10^4) and 'controller_avg_write_io_latency_usecs_vdisk…'(10^3) are also different. The ability to handle time-lag and scale variance is well-suited to analyze failure risks in a heterogeneous infrastructure.

AdeptDC's time series correlation ranking offers an efficient tuning capability which gives an opportunity for an operator to pass in the execution context by deleting one or two ranked entries in an ad hoc manner. Our algorithm will quickly recompute the inter-dependencies and update the ranking. As shown in Figure 21, an operator a can delete a ranked entry (Figure 21(a)) and recompute a new ranking quickly (Figure 21(b)).

Figure 20: Time series correlation ranking for a Nutanix Prism cluster

Figure 21: Tunable correlation ranking for a Portworx cluster.

(a) Before a row deletion. (b) After a row deletion

AdeptDC offers two more root cause analysis capabilities: one for offline impact analysis between two variables, as shown Figure 33 in the Appendix, and another for online dependency analysis between two sensors, as shown in Figure 34 in the Appendix.

Failure Prevention Recommendation

Most of the existing server remediation systems depend on runbook scripts and standard operating procedures. They essentially match a failure ticket with a historical failure, and then use the corresponding historical remediation. This strategy, however, is limiting because no two failures are identical in a distributed computing system with large degrees of freedom. To address this problem, AdeptDC features an AI-based recommendation engine to take failure prevention actions as quickly as possible.

AdeptDC classifies a failure based on the corresponding mitigation mechanism: one, online mitigation (defined as a failure mitigation that takes place without downtime), and another, offline mitigation (defined as a failure mitigation that requires an external action such as Replacement, Reimage, Reboot, Shutdown). An example of an online mitigation is the prevention of PROCHOT thermal throttling by adjusting environmental temperature setpoints, while that for an offline mitigation is the prevention of PROCHOT thermal throttling by replacing dead chassis fans. The choice depends on the operational context. To automate the mitigation type selection, AdeptDC offers a classification engine, as shown in Figure 22. AdeptDC's failure prevention recommendation engine works in sync with anomaly detection/failure prediction and root cause analysis. Anomaly detection and failure prediction together assist in identifying failure risks. Root cause analysis helps in narrowing the parametric space for the recommendation engine.

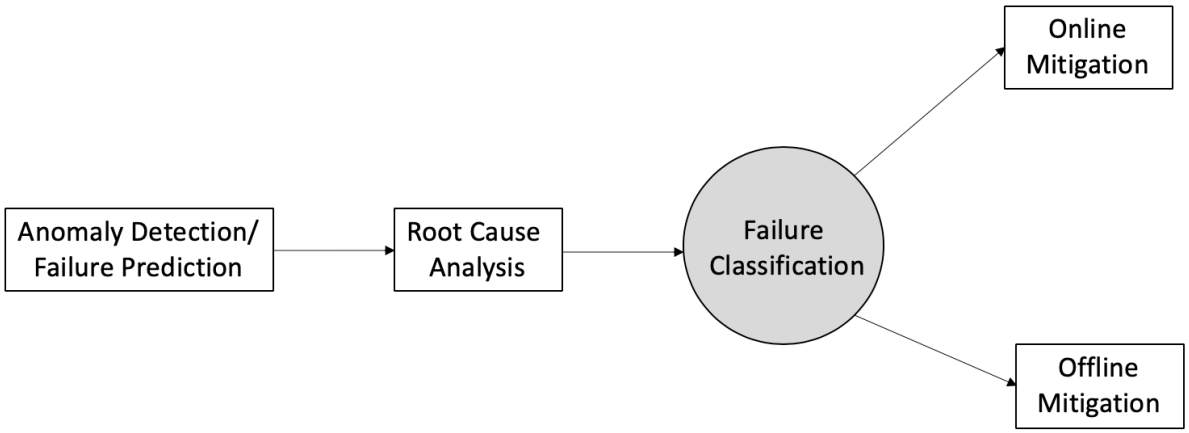


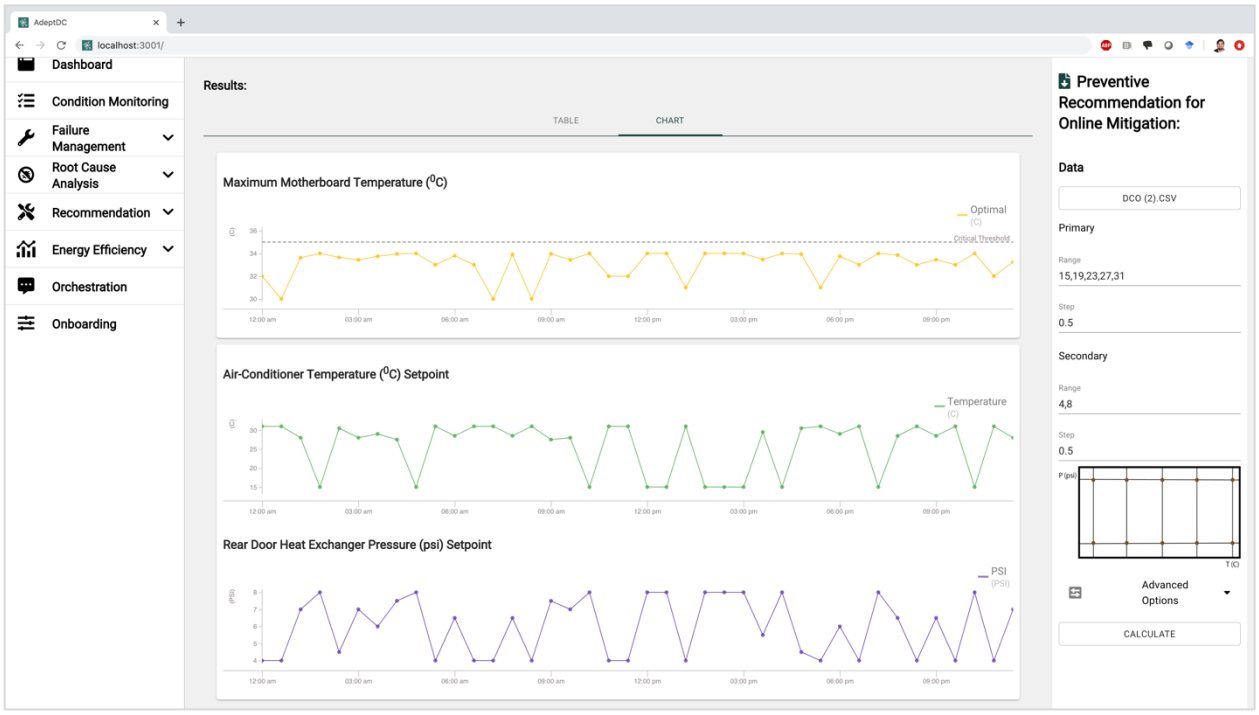
Figure 22: AdeptDC's failure classification workflow

Preventive Recommendation for Online Mitigation

AdeptDC's preventive recommendation for online mitigation combines an unsupervised pre-training algorithm and a reinforcement learning-based control algorithm. The unsupervised pre-training algorithm determines dynamic correlation patterns between a failure variable (e.g., processor temperature in the context of a PROCHOT warning) and a mitigation variable (e.g., cooling setpoints). Using the computed correlation pattern computed by the unsupervised pre-training algorithm, the reinforcement learning-based control algorithm computes optimal mitigation steps. The preventive recommendation engine is described in detail (Figure 35, Figure 36,Figure 37, Figure 38) in the Appendix.

Figure 23 shows AdeptDC's preventive recommendation engine in action for an online mitigation. In this context, the maximum motherboard temperature (^{o}C) or the critical temperature from a server fleet is chosen as the failure variable. The environmental cooling setpoints such as temperature setpoint (^{o}C) and pressure setpoint (psi) are two mitigation variables. The dotted line in the critical temperature chart represents a critical temperature threshold. AdeptDC's preventive recommendation engine continuously predicts optimal temperature setpoints (green line) and pressure setpoints (magenta line) for keeping the

maximum motherboard temperature (yellow line) below the critical temperature threshold with minimal possible energy consumption. The optimal cooling action can be implemented by interfacing with a building management system such as Tridium Niagara. The reinforcement learning algorithm we have used is completely general-purpose. [] can use this algorithm for intelligent virtual machine (VM) migration.



1. The maximum motherboard temperature can be considered a failure variable.
2, 3: Cooling setpoints which can be manipulated for the failure risk management They represent two mitigation variables.

Figure 23: Continuous recommendation for cooling setpoint adjustments to avoid overheating

Preventive Recommendation for Offline Mitigation

AdeptDC's preventive recommendation engine for offline mitigation uses a natural language processing-based analytics by leveraging historical failure data. Most server failures come with repair tickets for technicians. A repair ticket should be created with an attached recommended action to minimize operating overhead. In the context of hardware failure remediation, the examples of recommended repair action are Reboot, Reimage, Replace, and Shutdown. However, it is not easy to determine a recommended repair action using historical analytics. Given a large number of failure modes, often new failure types emerge without historical remediation records. These failure types typically rely on manual debugging. To reduce manual debugging time, AdeptDC's AI model can be trained from closed tickets in the past with raw text logs as features and can recommend repair action for a new ticket type. The recommendation is based on how similar tickets have been resolved in the past. Under the hood, AdeptDC uses

several state-of-the-art natural language processing (NLP) algorithms. Figure 24 shows AdeptDC's preventive recommendations for repair tickets for offline mitigation.

Repair Ticket	Recommendation
BIOS boot order incorrect	Replacement
NVDIMM-SW media change	Reimage
DRAM calibration failed	Reboot
Drive Capacity Degraded	Shutdown

Figure 24: AdeptDC's NLP-based preventive recommendation engine for offline failures

Orchestration

AdeptDC is capable of delivering several integrations to implement the remedial recommendations for different failure types.

Table 3: Possible implementation strategies

Failure Type	Orchestration
Component Overheating	Building Management System
Resource Starvation	Cloud Orchestration Platforms Such as VM Orchestration Platform
Network Saturation	Network Management System

Scalability

The scalability of any machine learning solution depends on the underlying algorithm and software architecture.

Algorithm

From an algorithmic standpoint, AdeptDC has adopted the following tactics to ensure scalability:

Unsupervised Pre-Training: For deep learning and sequence modeling, we use unsupervised pre-training algorithms to extract high quality features for improving model accuracy and efficiency. This unsupervised pre-training minimizes the dependency on labeled training data which is an expensive prerequisite for traditional machine learning platforms such as AWS SageMaker.

Polynomial Time Complexity: Most of AdeptDC's computational algorithms have polynomial time complexity. Therefore, they can efficiently scale even with a large volume of training data.

Dimensionality Reduction: AdeptDC's data compression algorithm is suitable for high-dimensional monitoring and log-data analysis. We offer more than 90% data compression with 1-5% data loss.

Transfer Learning: Model creation can be expensive. To avoid this predicament, AdeptDC has developed a transfer learning approach where the learning from one problem can be reused in a different but related problem. This approach will be very useful for training large neural networks for high-dimensional log-analysis, failure prediction, and recommendation engines.

Model Selection: Unlike other machine learning platforms, AdeptDC's AI/ML platform can automatically choose suitable training algorithms based on the variability pattern of the training data. Data with higher variability requires more resource-intensive models compared to a dataset with lower variability. For example, for low-variance data forecasting, AdeptDC will choose a simple forecasting algorithm such as Holt-Winters. For a high-variance data forecasting, however, AdeptDC will use a complex forecasting algorithm such as long short term memory (LSTM)-based sequence modeling.

Hyperparameter Tuning: Unlike other machine learning platforms, AdeptDC's automated hyperparameter tuning capabilities minimize operator overhead for model training and model maintenance. The pertinent examples for hyperparameters are learning rate, epoch size, batch size (for deep learning), train-validation-test split, window size (for sequence modeling, failure forecasting, and anomaly detection), number of clusters (for clustering algorithms), weights for ranking algorithms (for correlation ranking and dynamic alarm ranking), etc.

Machine Learning Model Transparency: Although AdeptDC performs model selection and hyperparameter tuning automatically, AdeptDC offers full visibility to the selected models and hyperparameters so that a user can easily override AdeptDC's selections. In fact, AdeptDC's anomaly detection, ranking and other predictions come with easy tunability features with which a user can easily provide feedback on the model accuracy at the UI. Based on the user feedback, AdeptDC's AI/ML framework quickly performs model retraining and inference update. This way AdeptDC offers significantly more visibility into the AI/ML computational processes compared to a traditional black-box AI/ML platform.

Software Architecture

Our machine learning algorithms primarily fall into two buckets: first, *Big Data Algorithms* which require a large volume of training data, typically on the order of a few GBs; and second, *Small Data Algorithms* which do not require a large volume of training data, typically on the order of a few KBs. While the deep learning-based models fall into the *Big Data Algorithm* bucket, the correlation ranking and other machine learning algorithms which compute and update inferences on the fly (e.g., failure domain clustering) fall into the *Small Data Algorithm* bucket.

Each bucket has its own unique scalability challenges which we address, as follows:

a. *Big Data Algorithm*: We use data warehouse solutions (e.g., Hadoop, AWS Redshift, Azure Synapse) to store our data and run our training algorithms as MapReduce frameworks. This gives us Horizontal scalability. The trained models are stored using an object storage service such as AWS S3 or Azure Blob Store. The inference engine is deployed as a stateless microservice which retrieves the trained models from the object storage service and returns output for the deployed input data, as shown in Figure 25.

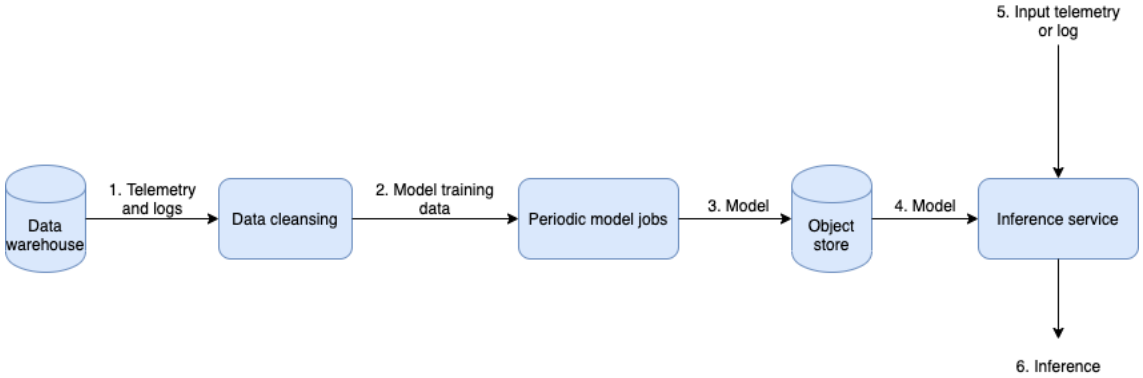


Figure 25: Architecture diagram for the machine learning pipeline for big data algorithms

b. *Small Data Algorithm*: We keep the data in key-value stores (e.g., Apache Hive, Azure Cosmos DB, Amazon DynamoDB, Cassandra) backed by a distributed in-memory cache (e.g., Memcached). Due to small data training data size, we can build models in near-real-time. Therefore, the model training and inference are put together in a single microservice, as shown in Figure 26.

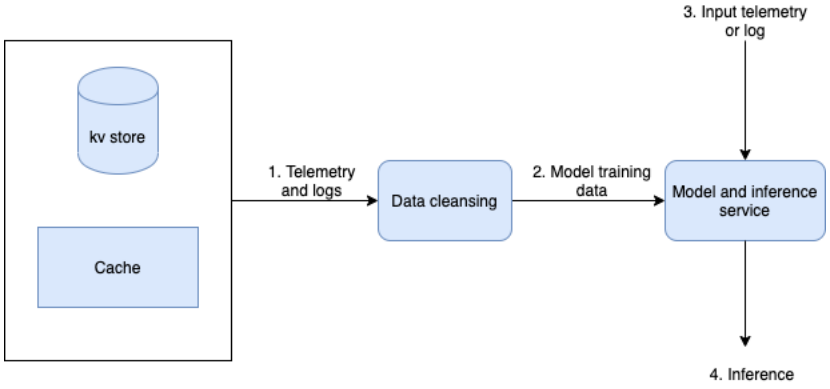


Figure 26: Architecture diagram for the machine learning pipeline for small data algorithms

In addition to these two buckets, there is a third bucket for hybrid algorithms. A hybrid algorithm combines big data algorithms with small data algorithms. A pertinent use case is failure forecasting with a timestamp in which a long-range forecasting algorithm (a big data algorithm) needs to be complemented by a small range filtering algorithm for local fluctuation detection (small data algorithm). A hybrid algorithm is also useful for incremental retraining. An architecture diagram for the machine learning pipeline for hybrid algorithms is shown in Figure 27.

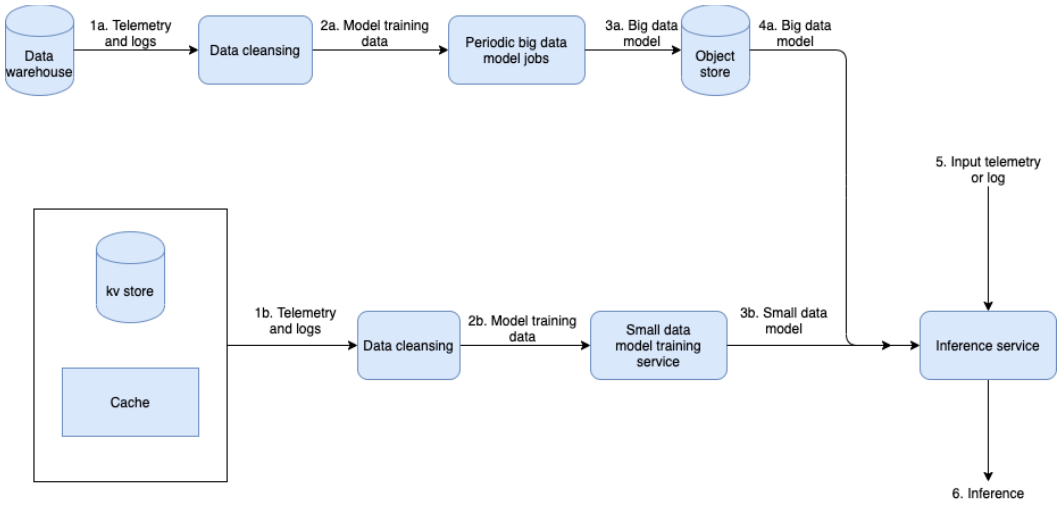


Figure 27: Architecture diagram for the machine learning Pipeline for hybrid algorithms

IV. Hand-Off Procedure at the End of the Project – Source code, binaries, documentations

We will offer a SaaS solution. The source code and binaries hand-off will be discussed separately as the engagement progresses.

V. Appendix

Machine Learning Pipeline

For anomaly detection/failure prediction, an ensemble approach will be used. For the ensemble approach, several time series learning algorithms such as long short term memory (LSTM), gated recurrent unit (GRU), Holt-Winter, and seasonal autoregressive integrated moving average (SARIMA) will be used to reach the optimal bias-variance trade-off point. Necessary pre-processing and post-processing (e.g., data cleansing, dimensionality reduction, feature engineering, etc.) will be conducted to improve prediction fidelity.

For auto-remediation, semi-supervised reinforcement learning will be used. An unsupervised algorithm will be used to build a correlation engine between a set of failure and mitigation variables. After modeling the correlations, a reinforcement learning-based control algorithm is used to identify the optimal remediation strategy to remediate a failure state. Figure 28 shows the architecture diagram for the solution which combines an anomaly detection/failure forecasting module with an auto-remediation module.

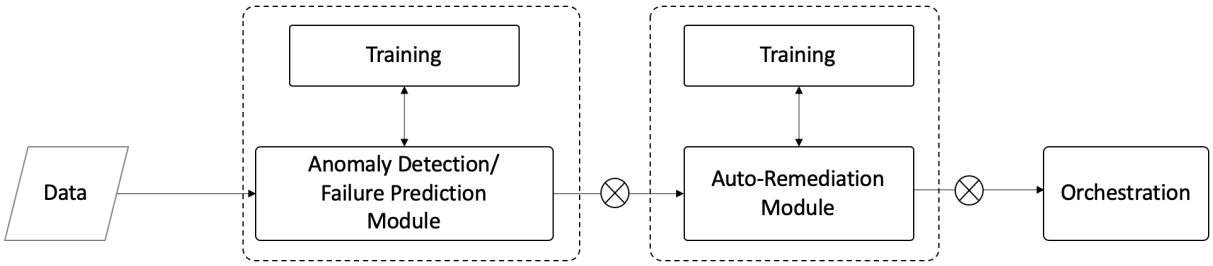


Figure 28: Machine learning pipeline for the proposed predictive maintenance solution

Human-Authorized AI Action for Operational Safety

There are several measures to ensure the safety of AI-enabled automated operations:

- Manual override to supersede any AI action.
- Automatic failover to a neutral state if the AI control system violates a safety constraint. The neutral state can be defined by rules and heuristics. We would also ensure smooth failover to prevent sudden changes in the system.
- We offer three different operational modes: enforcing, permissive, and disabled.
    - Enforcing: All remediation strategies are enforced.
    - Permissive: No remediation takes place. Only the remediation strategies are logged.
    - Disabled: All remediations are disabled.

Monitoring Sensor Health Scores

Figure 29: Sensor list from a DRAC server monitored by AdeptDC

Hierarchical Cluster Analysis

AdeptDC offers hierarchical clustering capabilities so that a user can easily group log files and keywords based on their based on their similarities. It supports drill-down of a corpus of log files for failure risk analysis. Figure 30 shows a dendrogram from AdeptDC's Hierarchical Clustering Analysis for a simulated dataset. This analysis suggests a group of {Disk, Motherboard} which means if Motherboard starts throwing off failure messages, then a user should run a failure risk check for Disk, too.

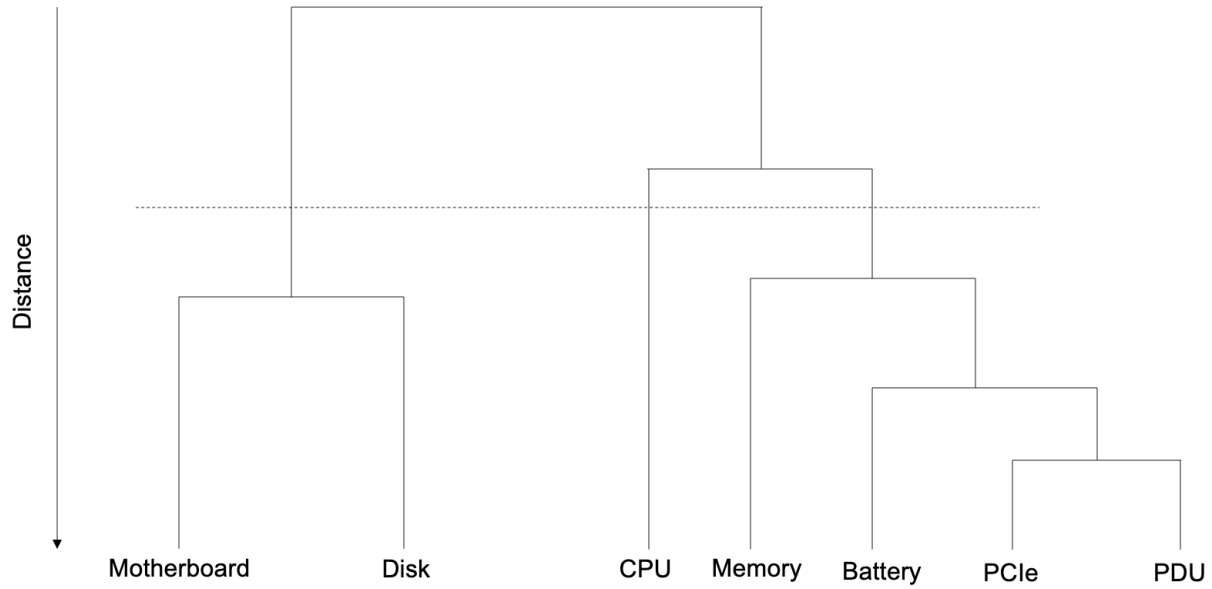


Figure 30: A simulated example for hierarchical cluster analysis for log files

Alarm Policy, Notification, and Dynamic Alarm Ranking

A distributed system often produces a high volume of isolated alarms, especially in the context of cascading failures. Even though isolated, these alarms are strongly correlated. AdeptDC offers a relevance ranking capability which ranks alarms in their order of relevance. It essentially converts a vanilla alarm into a contextual alarm which minimizes alarm fatigue. Figure 31 shows the results from dynamic alarm ranking for a Portworx cluster for a time period from January 1, 2020 4:11PM to January 11, 2020 5:11PM. The ranking algorithm has logarithmic time complexity and took 3 minutes and 20 seconds to rank 1178 metrics.

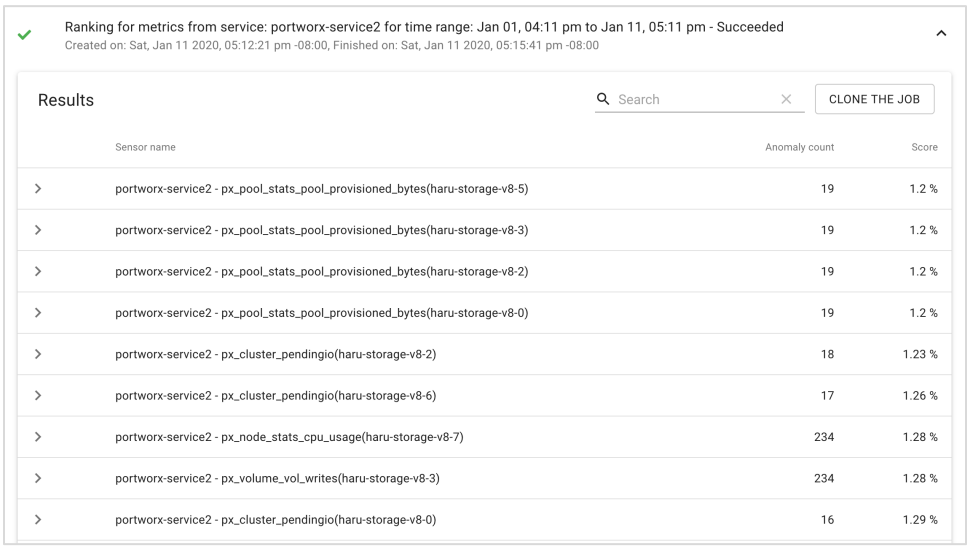


Figure 31: Dynamic alarm ranking for Portworx cluster metrics

As shown in Figure 31, each row of the ranking table consists of a pertinent metric or sensor name, anomaly count, and score. Our relevance algorithm computes the relevance score for each metrics, normalizes relevance scores using max-min normalizations, and shows them in an ascending order as shown in Figure 31. It is important to note that the anomaly count hardly has little bearing on the relevance ranking. As shown in Figure 31, "px_node_stats_cpu_usage" with an anomaly count of 234 has a lower ranking to "px_pool_stats_pool_provisioned_bytes" with an anomaly count of 19. Several metrics might have identical relevance scores. In such a scenario, a user will have an opportunity to employ their domain knowledge. AdeptDC's contextual AI software accepts user feedback on the fly and recomputes the ranking, as shown in Figure 32.

As shown in Figure 32, a user can delete one or more entries in the ranked list based on the dynamic user context and our algorithm will update the ranking. It not only deletes a single entry, but it also updates the ranking of the other elements. Essentially, our network model is capable of updating the complex interdependencies among different nodes.

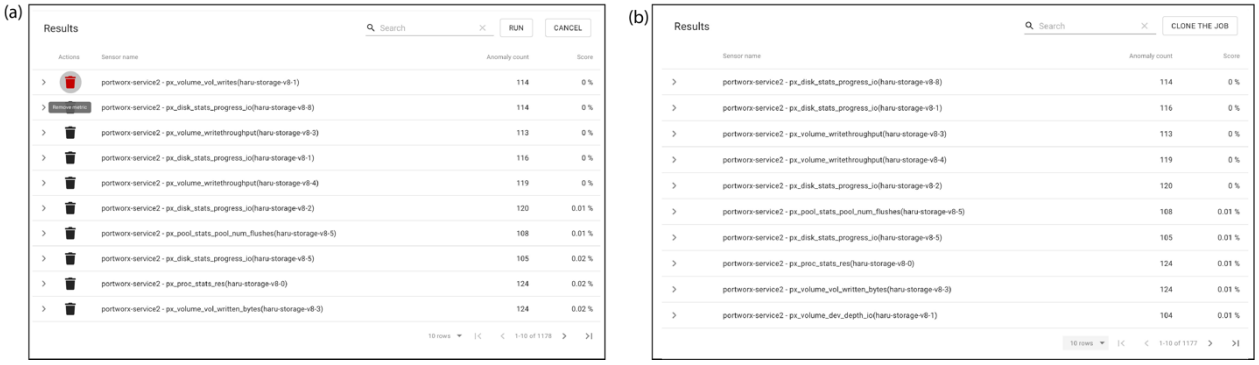


Figure 32: Tunable Alarm Ranking

(a) Alarm Ranking before a row deletion (b) Alarm Ranking after a row deletion

Root Cause Analysis

Impact Analysis

AdeptDC offers an impact analysis capability with which a user can rapidly compute the influence of one parameter on another. Figure 33 shows how we can predict CPU temperature (^{0}C) as a function of GPU utilization (%) with 95% confidence. The redline in Figure 33(b) shows the actual GPU temperature. Other lines represent predicted values for 20%, 40%, and 70% GPU utilizations.

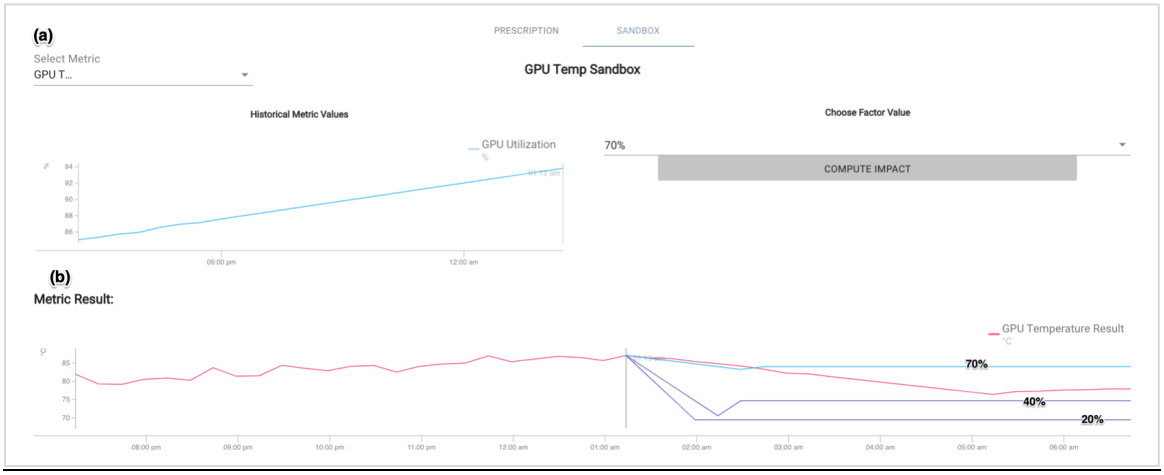


Figure 33: Interactive dependency mapping between GPU utilization (independent variable) and GPU temperature (dependent variable)

Sensor Fusion

We have developed a sensor fusion capability where one sensor data can be used continuously to predict another sensor value. Figure 34 shows how we can continuously predict GPU temperature (^{0}C) as a function of GPU utilization (%) with 95% confidence.

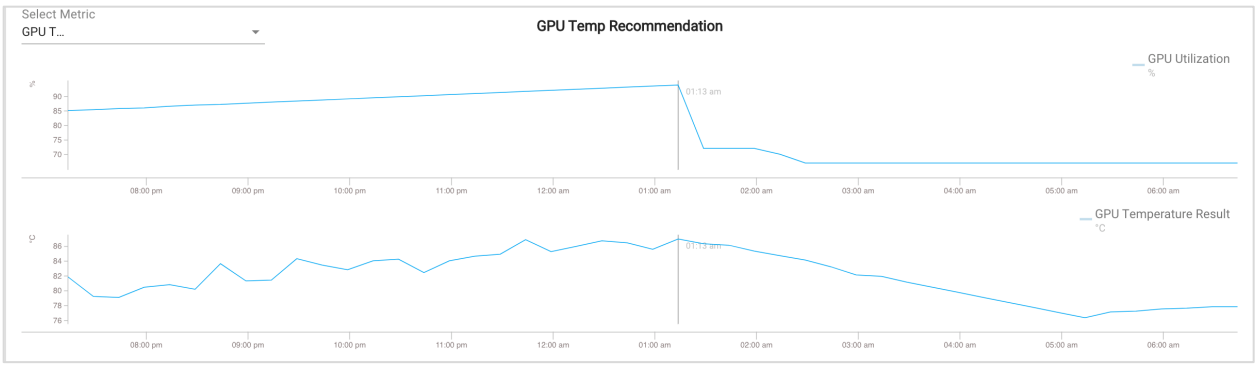


Figure 34: Dynamic GPU temperature (^{0}C) forecasting as a function of GPU utilization (%)

Laboratory-Scale Validation of Auto-Remediation Algorithm for Online Failures

Reversible failures (or, online failures) are defined as the failures that can be avoided by tuning some of the prevailing remediation variables. Examples of transient failures are processor overheating PROCHOT[30] failure which could be remediated by dynamic cooling setpoint adjustment and latency spike due to resource starvation. We model the auto-remediation of transient failures as an optimization problem with the objective function is to recover from the failure state within a reasonable timeframe with minimal operating cost and disturbance. As a solution methodology, we have developed a semi-supervised reinforcement learning algorithm. The general methodology for the auto-remediation algorithm is listed, as follows:

1. Our platform pulls telemetry data every minute.
2. The information is fed into a learning network to forecast the failure state.
3. If there is a near-term failure risk, the reinforcement learning-based control algorithm determines actions that minimize failure risks and satisfy user-defined constraints.
4. Our platform is capable of handling operator-driven supervisory control.

The effectiveness of our remediation algorithm is validated in the context of an overheating scenario. We use CPU temperatures as the process variable for the learning platform and distributed cooling setpoints as the remediation variables. As shown in Figure 35, the process variable for the validation case study is CPU temperatures for HP Proliant servers. There are 168 CPUs in the test rack D-5. The remediation variables include CRAC-1 temperature setpoint (^{0}C) and rear door heat exchanger (RDHx) pressure setpoint (psi).

First as shown in Figure 36, our time series clustering algorithm helps in grouping 168 CPU temperatures based on their variability patterns and, thereby, eliminates the necessity for brute-force troubleshooting. Each CPU was subjected to a simulated utilization profile, as shown in Figure 36(a). Each CPU temperature is modeled by a multi-dimensional time series with environmental factors such as cooling setpoints as parameters. As shown in Figure 36(b), the clustering algorithm groups multi-dimensional CPU temperatures into three classes and projects

[30] https://www.maximintegrated.com/en/glossary/definitions.mvp/term/PROCHOT/gpk/896

them into a two-dimensional plane for the visualization convenience. In Figure 36(c)-(e), each class is analyzed separately. For each class, the box plots of all CPU temperature time series are compared side-by-side. As shown in Figure 36(c)-(e), the members of each class exhibit similar averages and deviations. Evidently, Class-3 (as shown in Figure 37) shows the maximum failure risk with both the highest average temperature and a large number of outliers.

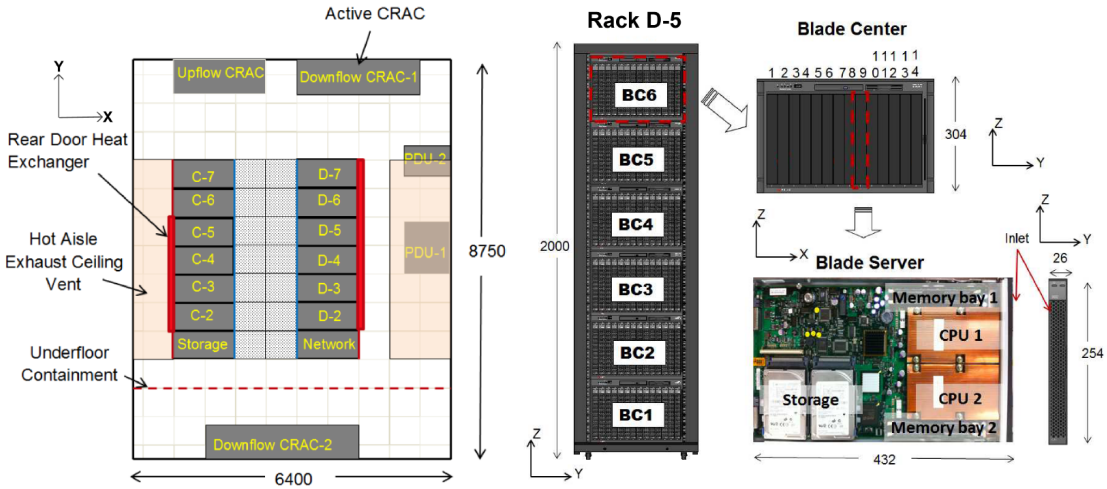


Figure 35: Laboratory-scale test setup for the auto-remediation algorithm

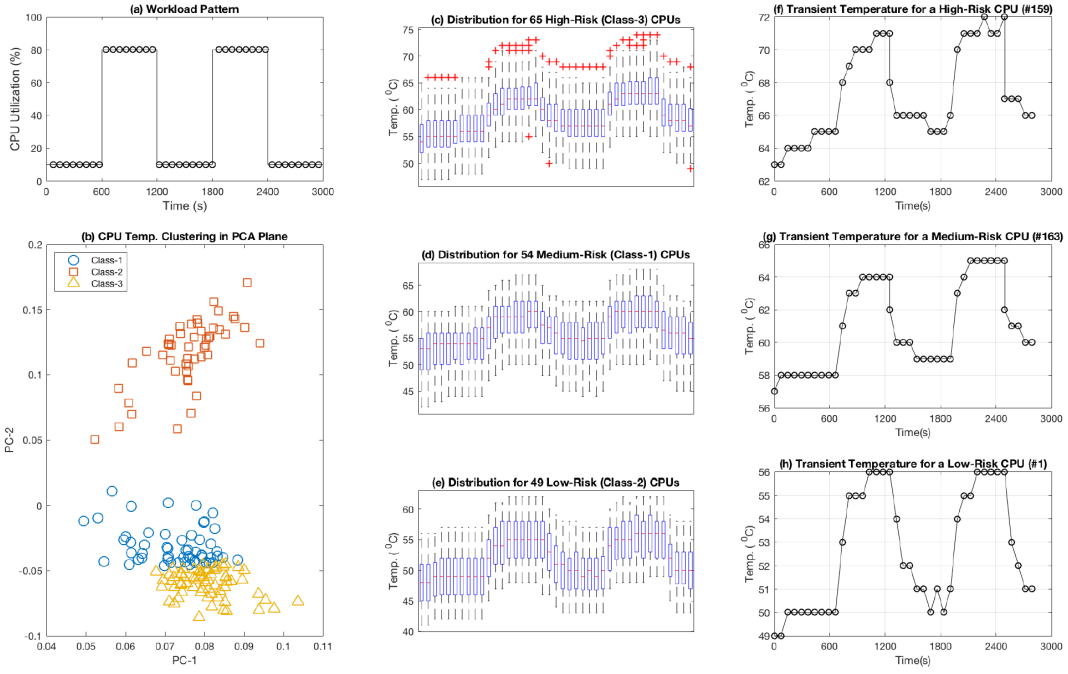


Figure 36: Dynamic failure risk triaging for CPUs.

(a) CPU workload benchmarking. (b) CPU temperature clusters. (c)-(e) CPU temperature distribution within a class. (f)-(h) Representative CPU temperature profile from each class.

Next we pick up two representative sensors from the riskiest class-3 CPUs. One of them is the temperature sensor for CPU#160 which shows the maximum temperature for class-3 CPUs. The second of them is the temperature sensor for CPU#127 which shows maximum volatility among all CPUs in Class-3. Incidentally, it was discovered that CPU#160 shows the maximum average temperature as shown in Figure 37 (a)-(b). On the other hand, CPU#127 shows the maximum standard deviation, as shown in Figure 37 (c)-(d). Both of these CPU temperatures stayed above 65 ^{0}C, which is the manufacturer-specified reliability limit.

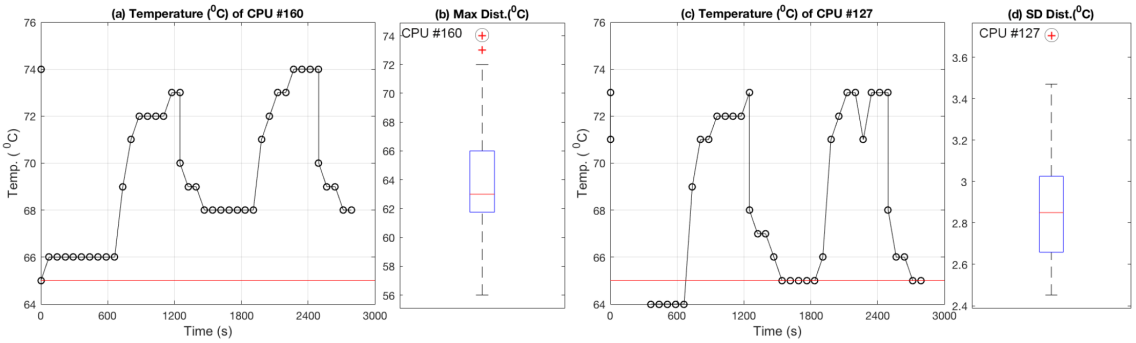


Figure 37: Monitored CPU temperature sensors for the validation case study.

(a) Transient temperature for CPU#160 which shows the maximum temperature among the test CPUs. (b) Maximum temperature distribution for all Class-3 CPUs (riskiest CPU class). (c) Transient temperature for CPU#127 which the maximum volatility among all the test CPUs. (d) Volatility distribution for all Class-3 CPU (riskiest CPU class).

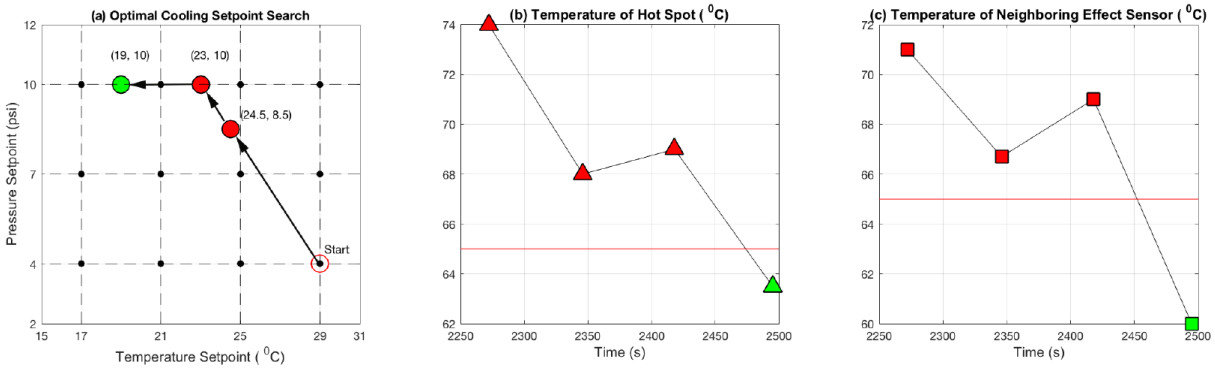


Figure 38: Auto-Remediation sequence for CPU overheating.

(a) Cooling setpoint adjustment sequence. (b) Transient temperature for CPU#160. (c) Transient temperature for CPU#127.

Figure 38 shows the overheating remediation action. A worst-case scenario was modeled with the remediation sequence starting from 2,722 s, when CPU#160 reached its maximum temperature with the most power-intensive setpoints of (29 0C, 4 psi). The proposed algorithm tool takes three steps to resolve the overheating problem. The resolution period lasts for 223 s, involving a cooling setpoint adjustment sequence of: (29 ^{0}C, 4 psi) → (24.5 ^{0}C, 8.5 psi) → (23

^{0}C, 10 psi) → (19 ^{0}C, 10 psi). At the end of the resolution sequence, the temperature of CPU#160 reaches 63.5 ^{0}C and that for CPU#127 reaches 60 ^{0}C—both below the safety limit.

Our algorithm is workload-optimized. We have validated the robustness of the underlying algorithms under different operating conditions with different workload shapes (as shown in Figure 39).

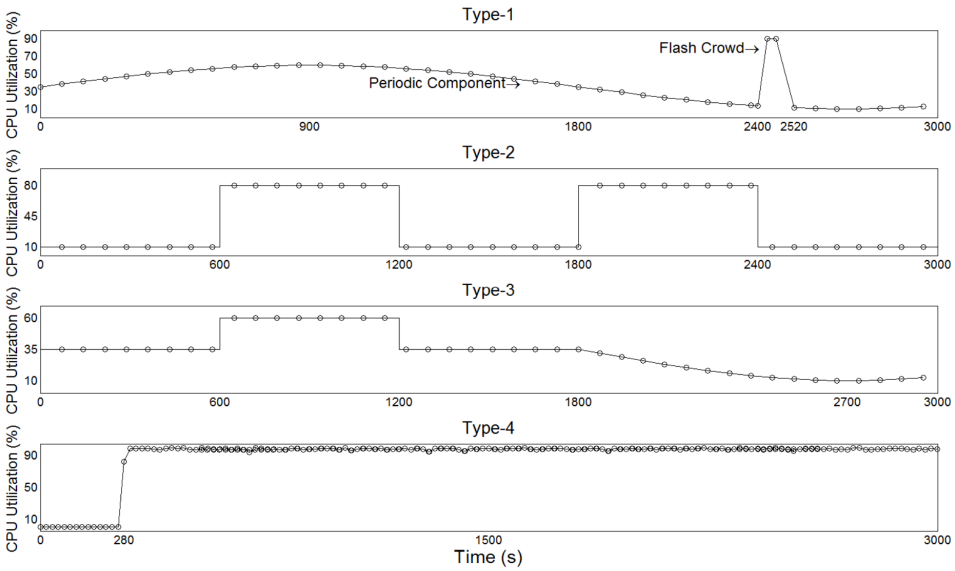


Figure 39: Different benchmarking workloads for AdeptDC's online remediation algorithm[31]

Tunability

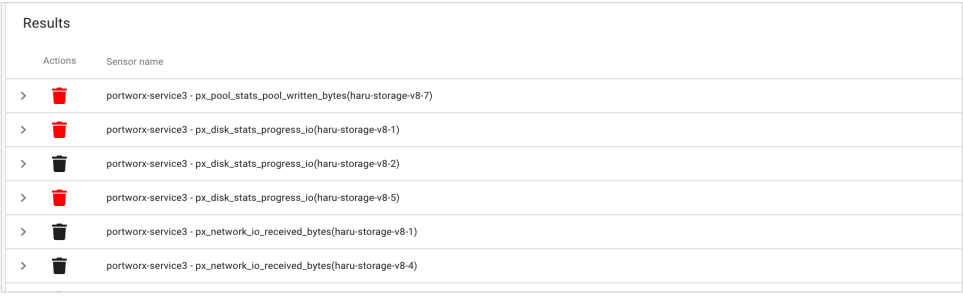


Figure 40: Tunable ranking capability

Our anomaly detection, correlation ranking, and dynamic alarm ranking capabilities come with a tunability capability with which a user can quickly pass feedback and update predictions. Figure 40 shows our tunable correlation ranking capability. We have already discussed the tunable anomaly detection (Figure 9) and tunable dynamic alarm ranking (Figure 21).

[31] https://patentimages.storage.googleapis.com/1b/ee/39/6bf812004e6d81/US10439912.pdf

AutoML Features

Designing neural nets is extremely time intensive and requires an expertise that limits its use to a smaller community of engineers. That's why AdeptDC has created an approach called AutoML which abstracts complicated hyperparameter tunings for the operators. AdeptDC's AutoML determines and tunes all hyperparameters automatically. In addition, we also offer overriding capabilities so that the operators can update the model parameters based on their specific requirements (Figure 41,Figure 42).

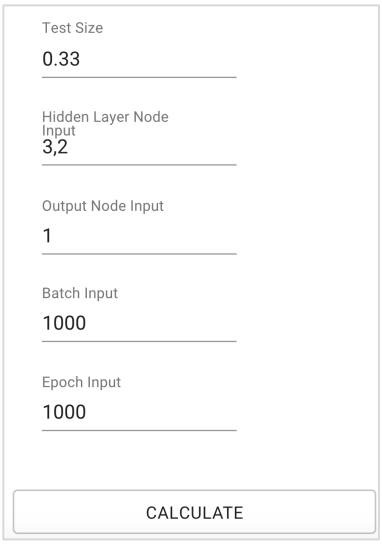


Figure 41: User-defined hyperparameter specifications (relative test fraction, hidden layer, batch size, epoch) for a deep neural network

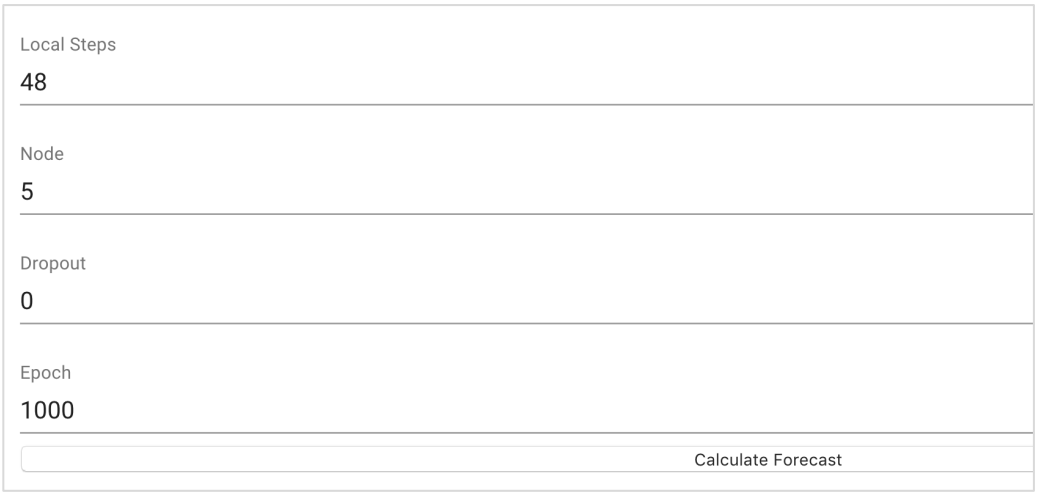


Figure 42: User-defined hyperparameter specifications (local steps, node, batch size, epoch) for a long short term memory (LSTM) network