ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 6 Issue 3 July 2026
Paper Id: JETA-V6I3P110/ Page No: 84-92

Original Article

Intelligent Failure Prediction in Enterprise Systems Using Behavioural Pattern Modelling

Pradip Baral
West Bengal University of Technology, Kolkata, India

Received Date: 12 August 2026 Revised Date: 17 August 2026 Accepted Date: 22 August 2026

Abstract: *Intelligent failure prediction in enterprise systems has advanced, from threshold-based failure predictions to modelling behaviours across logs, metrics, traces, storage telemetry, workload rhythms and signs of service interactions. The review is limited to peer-reviewed papers published in journals or conference proceedings from the year 2015 to 2024 and emphasizes on the following areas: operational anomaly detection (OAD), predictive maintenance (PM), system-log reliability analysis (SLR), multivariate time-series modelling and failure oriented interpretation. Reviewed evidence demonstrates that having capability to integrate recurring operational sequences, temporal deviations, seasonal key performance indicators, degradation trajectories and component-level health signals in the same behavioural model contributes to an improvement in early warning capability, because they are normally considered as separate warnings. The literature, however, is spread across cloud computing, microservice observability, industrial prognostics, hard-drive failure prediction, and general anomaly detection. Persistent limitations include weak cross-system generalization, limited labelled data, unstable log templates, limited causal explanation, missing failure labels, and insufficient evaluation under real enterprise drift. More interpretable, topology-aware, and/or temporally calibrated models that relate deviations in behaviour to concrete actionable failure risk instead of the post hoc flagging of anomalies are needed in the field.*

Keywords: *Anomaly Detection, Behavioural Pattern Modelling, Enterprise Systems, Failure Prediction, Log Analytics, Multivariate Time Series*

I. INTRODUCTION

The failure prediction problem for enterprise is another important current problem of reliability because of the usage of distributed applications, cloud platforms, storage arrays, data pipelines, industrial control layers, and service-management processes in an enterprise, all of which are dynamic in nature. In this context, failure is not likely to come as a sudden stop. It usually comes in one of the following forms: repeated sequence deviations, metric co-movement, workload seasonality, access rhythms, resource saturation, queue accumulation, component degradation. Automated log analysis is thus also an important task for reliability engineering, since execution paths along with exception events and failure-prone events hidden in the logs are not necessarily identifiable from scalar monitoring [1]. Intelligent failure prediction in enterprise systems with behavioural pattern modelling focuses on a specific aspect of anomaly detection, namely, it is about using learned behavioural regularities for prospective estimation of failure risk, instead of detecting abnormal observations.

The current research landscape is affected by changes in architecture and operational design. A microservices architecture has introduced agility in deployment, but runtime dependencies, small failures, cascading failures, and observability issues have been introduced between services [2]. Local resource exhaustion, traffic fluctuations on the upstream service, latency on the downstream service, configuration drift or hidden contention on the infrastructure can all account for a metric anomaly in one service. The relevance of behavioural pattern modelling is clear as enterprise failure prediction should be able to identify such context dependent and time ordered signals. Time-series analysis research offers means for seasonality, trend, subsequence deviation and distributional change; however, there are not many models to consider that relatively stable data-generating conditions are typically not found in an operational platform [3]. Although significant advances have been made in deep anomaly detection research to make predictions, reconstruction, density estimation, or self-supervised objectives more expressive, there is a recurring critique from the literature on issues of interpretability, calibration, and robustness to distribution shift [4].

Given the interdisciplinary nature of the topic, software reliability engineering and machine learning, operations research, storage dependability, industrial prognostics, and service management all intersect. While predictive maintenance literature has shown that degradation behaviour can be modelled from sensor and operational signals, enterprise IT systems present other challenges; some failure labels have multiple interpretations; incident tickets may come before the degradation within the enterprise, and the actions taken to remediate the incident may alter the degradation behaviour prior to a failure being observed. Hence, enterprise failure prediction is not possible by simply taking the accuracy score. It involves issues such

as warning lead time, cost of false alarm, alert fatigue, close location to the root cause, and the significance of the modelled pattern in operation.

The literature is scattered, creating a need for review. Some studies are centered around cloud application operations, with the use of logs and metrics, others around disk health prediction via recurrent networks, some are streamed anomaly detection, some are deep time-series models and some are enterprise concepts for predictive maintenance that are not directly applicable to cloud. A search for studies directly addressing this exact topic shows that the literature is relatively limited. Therefore, this review is restricted to specifically peer-reviewed articles published in journals clearly related to the main mechanisms of this topic, namely the extraction of evidence from behaviour, the modelling of time, the interpretation of anomalies to failure, the prediction of degradation and the monitoring of the operations of an enterprise for operational relevance. This article is intended to critically review and discuss past and current usage of behavioural pattern modelling in intelligent failure prediction applied to enterprise systems, and to suggest how potential rigorous adaptations of the behavioural pattern modelling approach could be applied to the intelligent failure prediction problem for enterprise systems. The article compares studies on the basis of their type of study evidence, type of modelling, reported results, and operation restrictions. It also provides a small conceptual and visual sketch for a successful understanding of the integration of logs, metrics, traces, storage-health indicators and temporal signatures to failure-oriented prediction systems. This review is not intended as a general survey of anomaly detection, cloud computing, or AI; rather, it focuses on predicting enterprise-system failure from behavioural patterns.

II. LITERATURE REVIEW

The relevant literature on intelligent failure prediction in enterprise systems can be classified as follows: (1) Operational behaviour mining, (2) Temporal anomaly modelling, and (3) Degradation-oriented prognostics. Behavior mining for operation is done at runtime, based on application events, system logs, and runtime metrics. Farshchi et al. analyzed cloud application operations using correlation between log evidence and cloud metrics, and demonstrated that operational failure is easier to detect when multiple behavioural signals are correlated together as opposed to being monitored independently [6]. This is significant because there are several partial views of failure in enterprise systems. A log sequence can show that the program is executing in an unusual way, a metric trend can show that resources are being over-used, and if both are correlated to the same behavioural episode, their predictive power increases.

Table 1: Summary of key Findings

Ref	Focus	Key Findings
[6]	Cloud application operations	Log–metric correlation improved operational anomaly interpretation and reduced dependence on isolated alert streams.
[7]	Predictive maintenance classification	Multiple-classifier designs showed that failure prediction must account for maintenance cost and decision timing.
[8]	Storage-system health behaviour	Recurrent health-state modelling captured temporal degradation signatures before hard-drive failure.
[9]	Streaming anomaly behaviour	Online unsupervised modelling detected deviations in changing temporal streams without dense labels.
[10]	Scalable KPI forecasting	Decomposable trend and seasonality modelling supported interpretable baseline construction for enterprise metrics.
[11]	Deep temporal anomaly modelling	Convolutional sequence learning detected abnormal subsequences in multivariate time-series behaviour.
[12]	Machine-health monitoring	Deep architectures improved feature learning from complex degradation signals but increased interpretation difficulty.
[13]	Deep remaining-useful-life estimation	Convolutional prognostics linked temporal degradation patterns to estimated failure horizons.
[14]	Health prognostics review	Prognostic performance depended strongly on data acquisition quality, health indicators, and operational context.
[15]	Data-driven predictive maintenance	Surveyed data-driven maintenance methods showed recurring gaps in transferability and deployment validation.
[16]	IoT time-series anomaly detection	Edge and sensor anomaly studies highlighted streaming constraints relevant to enterprise observability.
[17]	Deep and shallow anomaly detection	Unified comparison showed that representation learning improves flexibility but complicates reliability guarantees.

Several studies provide algorithms for streaming signals, seasonality and high-dimensional deviations. Ahmad et al. showed that unsupervised streaming anomaly detection systems can learn temporal patterns when they change without the need for dense labels for failures [9]. Taylor and Letham demonstrated that decomposable forecasting can be very useful for large-scale time-series seasonality and this is directly applicable to enterprise KPI baselines, as service demand can be very regular in terms of calendar and business-period rhythms [10]. In the same vein, deep learning algorithms like DeepAnT can be used to learn local temporal structure based on convolutional forecasting and reconstruction tasks [11]. These models offer valuable ability to identify deviations, but the presence of an anomaly does not imply failure. A deviation makes it into the failure-relevant domain when it is associated with risk, lead time, system context or degradation state.

Degradation-oriented prognostics brings in a different valuable view. Susto et al. presented predictive maintenance as a classification problem with operational cost constraints, noting that the beneficial impact of predictive maintenance is not just its classification accuracy but when it happens [7]. Xu et al. used recurrent neural networks to model the health states of hard drives and showed that sequential telemetry can detect the patterns of hard drive degradation prior to their physical failure [8]. This evidence is furthered by the machinery-health reviews and deep prognostics studies demonstrating the ability to map behavioural sequences to failure horizons through remaining useful-life modelling [12]–[15]. However, direct transfer to enterprise systems is limited because software-driven failures often arise from interaction effects, configuration changes, workload bursts, code releases, and dependency chains rather than gradual component wear.

As it can be seen in Table 1, the evidence base for component mechanisms is robust, but weak in the case of fully integrated enterprise failure prediction. While the cloud-operations literature is closest to the target problem, there is a lack of open-source examples and low level of reproducibility in enterprise studies. Predictive maintenance research offers ideas to help make decisions, but many assess physical equipment instead of software-driven enterprise actions. Time-series anomaly studies offer methodological sophistication, but are usually more anomaly-based than failure-based in their definition of outcomes.

Another restriction is with regard to the unit of behaviour. Single sensors, single KPIs or single components are modelled in some studies, while enterprise failure prediction typically involves modelling the multi-entity behaviour across services, queues, storage devices, deployment events and remediation actions. Cook et al. highlighted the need to address the challenges of time-series anomaly detection in networked sensor environments, such as volume, heterogeneity, and contextual variability of data [16]. Similarly, Ruff et al. demonstrated that the assumptions of normality, representation, and decision boundaries are significantly different among anomaly detection methods [17]. These assumptions are important because there are a lot of “normal abnormalities” in enterprise systems, such as scaling events, batch jobs, peak traffic, and planned deployments. A model that sees all rare behaviour as risk may be correct based on benchmark scoring but it may be counter-productive operationally.

In conclusion, there is a large body of literature that shows the potential of behavioural pattern modelling, and there is also a large gap in the field: there is no mature and well-established framework in the literature that maps the behaviour deviations into the probability of failure of the enterprise with interpretable lead time, topology context, and intervention value, established in a journal paper. Multimodal learning, temporal abstraction, and decision-aware evaluation are the three areas of strong study. The greatest limitation in the literature is not that there are no algorithms, but rather the lack of a strong correlation between algorithmic anomaly scores and failure management in complex enterprise environments.

III. METHODOLOGY

The modelling of behavioural failures for enterprise failure prediction can be broken down into a series of conceptual steps: evidence capture to operational decision. There are inconsistencies in the degree of evidence in reviewed studies, the goal of the modelling and how failures are interpreted. Log-oriented approaches convert textual events to templates, sequences, counts or embeddings; metric-oriented approaches model seasonality, residuals, correlations, or multivariate subsequences; prognostic approaches predict health state, or remaining time to failure. The difficulty of the conceptual issue is that each one of them reflects a different facet of enterprise behaviour. Logs are the semantics of execution, metrics are resource and performance dynamics, traces are dependency latency, and storage or device telemetry is component degradation. These views can be used in conjunction with each other and if they are aligned across time and risk episodes around the time of the alert, failure prediction improves.

Forecasting based approaches estimate what is expected to occur and alert to any deviation from the expected behaviour. The benefit, is interpretability, particularly if seasonality and trend are evident, such as in service traffic or KPI baselines [10]. However, these approaches have limited sensitivity to complex multivariate interactions. Deep learning models based on reconstruction, such as convolutional and autoencoding models, learn a compact representation of normal behaviour and identify high reconstruction error [11, 17]. These methods can capture more subtle deviations in multivariate data, but

might yield scores with little meaning for interventions. Log, storage telemetry, operational event streams are examples of data that can be processed with recurrent and sequence models, which have been shown to model temporal dependency and degradation [8]. Recurrent models can, however, suffer from overfitting local sequence conventions, and can suffer from degradation in software release changes and/or log-template instability.

Table 2: Method comparison

Ref	Method	Strengths	Limitations
[6]	Correlation of cloud logs and metrics	Connects semantic events with resource behaviour in operational incidents.	Depends on consistent logging and aligned timestamps.
[8]	Recurrent health-state prediction	Captures sequential degradation before storage failure.	Best suited to component telemetry rather than dependency-induced service failure.
[9]	Online unsupervised streaming detection	Adapts to evolving temporal behaviour with limited labels.	Produces anomaly evidence, not direct failure probability.
[10]	Decomposable trend-seasonality forecasting	Interpretable KPI baselines for calendar-driven enterprise workloads.	Limited modelling of nonlinear cross-metric interactions.
[11]	Convolutional time-series anomaly detection	Learns local temporal patterns without manual feature engineering.	Requires careful thresholding and context-specific validation.
[13]	Deep convolutional prognostics	Links degradation features to remaining-useful-life estimates.	Assumes measurable degradation trajectories.
[16]	Streaming time-series anomaly methods	Addresses scale, latency, and online monitoring constraints.	Often lacks enterprise incident semantics.
[17]	Deep representation-based anomaly detection	Handles complex high-dimensional behaviour.	Interpretation and calibration remain difficult.
[18]	Deep anomaly detection taxonomy	Clarifies one-class, reconstruction, and prediction objectives.	Method taxonomies do not resolve deployment validity.
[19]	Fault-injection dependability assessment	Generates controlled failure evidence for evaluation.	Artificial faults may not match production behaviour.

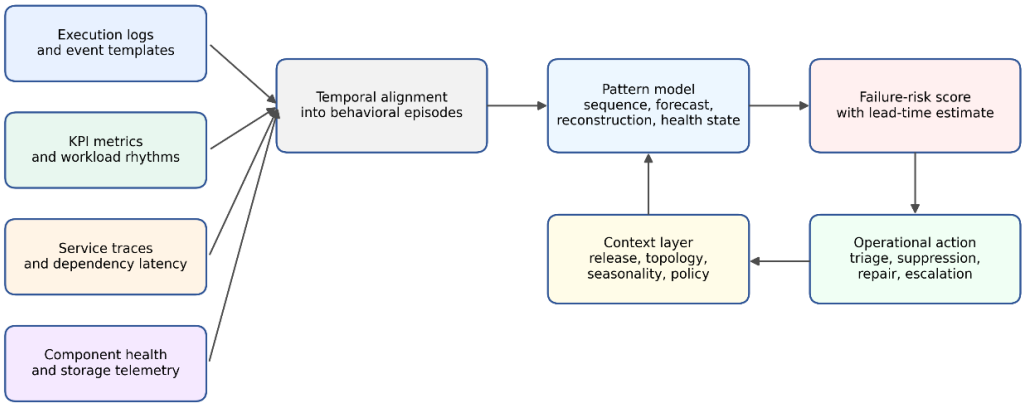

Figure 1: Behavioural Evidence-To-Failure Prediction Framework

A topic-based conceptual model is presented in Figure 1 that links together the different processes of evidence with the outputs of failures-oriented predictions. It is important because literature reviewed may optimise intermediate anomaly objectives, whereas enterprise operations call for risk, lead time and actionability. In this figure, see how enterprise logs, metrics, traces and component telemetry can be mapped to the behaviour episodes before you score the model and make operational decisions. The framework identifies that if temporal risk, context, and intervention logic are introduced, then the anomaly detection turns into failure prediction. The conceptual structure was designed on the basis of the examined Journal articles of the following concepts: log reliability analysis, cloud operations, temporal anomaly detection and predictive maintenance [1], [6], [8], [10], [17].

The framework also provides an explanation for why it is not sufficient to make comparisons between one method. The performance of the forecasting model can be good for KPI which are stable, but not for semantic log anomalies. A recurrent model can have some data being degraded, but it cannot account for failure of microservices. Fault-injection techniques can produce a labelled failure evidence; but, artificial faults can produce a purer signal than real ones under production [19]. Thus, the development of methodology should go hand in hand with the corresponding class of behaviour and the validation of the prediction using operationally meaningful results.

IV. DISCUSSION

Modelling behaviour as structured temporal evidence has been reported as the most effective approach to intelligent failure prediction. In all cases of failure-related information, it is not always found in one observation, be it in cloud operations, storage systems, streaming data or predictive maintenance. Farshchi et al. demonstrated that combining log and metrics makes the operational anomalies more interpretable, since failures tend to appear as a combination of semantic and resource-pattern anomalies [6]. Xu et al. demonstrated that health signals could be modelled sequentially for better inferences of failure risk for hard drives, as single measurements could be healthy but the trajectory unhealthy [8]. In enterprise environments, the workload patterns are constantly changing, and streaming anomaly detection needs to have this property; Ahmad et al. showed that streaming anomaly detection can capture changing temporal regularities [9]. These results highlight the significance of behavioural context—however, not every behavioural model has the same operational value.

One of the key trends is to move from static classification to temporal and representation based modelling. In [7], predictive-maintenance classifiers were shown to be useful in scenarios where not only the classification itself but also the time of making the decision and the cost of the decision were of concern. This was extended to learn latent degradation features from sequential data through deep temporal models and convolutional prognostics [11, 13]. However, this same trend brings in a contradiction. Richer representations learned by models can help them identify complex behavioural irregularities, which can also make them harder to explain and can hamper response efforts. Both Pang et al. and Ruff et al. acknowledged that there are problems with interpreting, calibrating, and transferring features with deep anomaly detection, in addition to its potential to enhance feature learning [4] [17]. These limitations are not secondary in the field of enterprise failure prediction. A model that yields an uncalibrated risk score that has no behavioural explanation could introduce operational noise and not decrease outage risk.

Table 3: Results Comparison

Ref	System / Context	Metric / Basis of Comparison	Outcome
[6]	Cloud application operations	Log–metric anomaly interpretation	Heterogeneous evidence improved incident-level interpretation.
[7]	Industrial predictive maintenance	Maintenance-oriented classification	Classifier selection depended on intervention cost and failure timing.
[8]	Enterprise storage devices	Sequential health-state prediction	Recurrent modelling captured degradation before disk failure.
[9]	Streaming telemetry	Online anomaly scoring	Adaptive temporal modelling detected deviations without labelled failures.
[10]	Large-scale operational KPIs	Forecast residual baseline	Trend and seasonality decomposition supported interpretable expected behaviour.
[11]	Multivariate time series	Forecast and anomaly error	Convolutional temporal learning detected abnormal subsequences.
[12]	Machine-health monitoring	Deep feature learning evidence	Deep models improved representation of nonlinear degradation behaviour.
[13]	Prognostic sequences	Remaining-useful-life estimation	Deep convolutional models connected temporal features to failure horizon.
[14]	Prognostics literature	Data-to-health-indicator pathway	Prediction quality depended on health-indicator design and acquisition context.
[16]	Networked time-series systems	Streaming anomaly constraints	Online scalability and heterogeneity shaped deployable detection quality.

The outcomes reported in or available in the literature from the studies reviewed are compared in Table 3. The comparison shows that “success” is very different in different contexts. Some studies have shown better anomaly detection,

others have focused on a general prognostic horizon, while others have given methodological taxonomies without deployment evidence. This variation makes it difficult to rank the evidence directly but facilitates identifying the contribution of evidence stream to enterprise failure prediction.

The evidence-distribution view of the literature reviewed is displayed in Figure 2. It reveals that metric and temporal-sequence evidences are the most prevalent in the journal corpus while enterprise-specific combinations of logs and topology together with incident outcomes are less prevalent. The bar chart sorts articles into categories based on the primary behavioural evidence type featured in each of the articles reviewed. Coverage of temporal metrics and prognostic sensor behaviour is good, and coverage of topology-aware enterprise incident modelling is lesser. All the references [6] – [20] were manually classified based on the primary evidence type dealt with in every article.

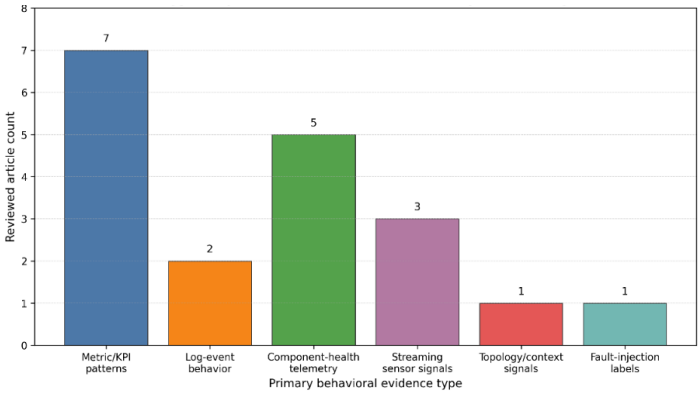

Figure 2: Evidence Types Represented in the Reviewed Journal Corpus

Figure 2 is important as it indicates a lack of balance in research. However, there is a lot of metric-centric work, since time-series data is easy to standardise across different domains, while the semantics of logs, and their topology context, are harder to publish reproducibly. However, enterprise failure prediction often calls for just those hard signals. An anomaly in CPU utilisation could be a normal period within a planned batch window, and an unusual order of log events following a deployment might signal a service issue before the aggregate KPIs begin to drop. Therefore, a behavioural model optimized only on metrics might not be complete.

A second trend is the separation between anomaly detection and failure prediction. Anomaly detection typically is expressed as a deviation from normal behaviour while failure prediction is the assessment of the likelihood of the deviation becoming a failure within an actionable horizon. This separation is reflected in the literature. Streaming anomaly models can detect unusual states [9] and deep time-series models can learn abnormal subsequences [11] but the output of these models does not necessarily include the probability of failure, the expected lead time or the remediation priority. Remaining useful life and predictive-maintenance research is more closely related to failure horizons [7], [13], [14] but many of them rely on degradation processes that are more predictable than interactions with enterprise software. Practical implication: Hybrid objectives are required for enterprise failure prediction: Anomaly score, Trajectory slope, Entity criticality, Dependency location, Historical incident association.

The comparison of methods is done based on four enterprise relevant evaluation dimensions in the form of an overview diagram as shown in figure 3. These values are not benchmark values but interpretive ratings based on the reviews of the studies. The comparison emphasises that there are no overwhelming advantages for any method family on all dimensions. The radar chart is a comparison of four method families in terms of label need, temporal depth, explainability and deployment fit for enterprise failure prediction. For the label-dependence dimension, higher values indicate lower dependence on labelled failure data [6]–[18].

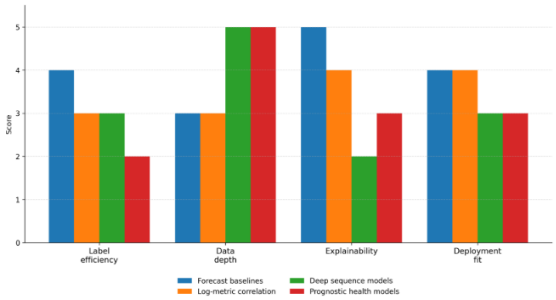

Figure 3: Comparative Method Profile for Behavioural Failure Prediction

The radar comparison is helpful for understanding an important research question. Forecast baselines are interpretable and deployable but have a limited capability in modelling service interactions. Explanation is weak, and deep sequence models give temporal depth. Log–metric correlation is a functional correlation, which depends on the consistency of instrumentation. Failure-horizon logic is provided by prognostic health models, but it does not transfer well to software-defined systems. Therefore, a mature enterprise failure-prediction model requires architectural integration rather than replacement of one method by another.

A third trend is the evaluation. While many anomaly studies report detection scores, enterprise operations demand warning lead time, class imbalance precision of alerts, root cause localization, stable across releases and minimizing the impact of an incident. There is a long history of literature on predictive maintenance that indicates that it is economically meaningful only when the prediction is timely enough to take action and not so early as to incur unnecessary action [5, 7, 15]. The more difficulty is added when remediation censors failure as is done in enterprise systems. For instance, when a failure outcome is automatically rolled back by an automated rollback, that failure outcome will no longer appear in the system, and the original behavioural precursor will be underlabeled. This means that the training data is biased and that the success rate of the prevention seems like failure. This is not an issue that has been completely answered in the literature.

A fourth trend is the domain context. There are also different definitions of anomalies for point anomalies, contextual anomalies, collective anomalies and subsequence anomalies [3], [16] as revealed by time-series surveys. Contextual Anomaly Interpretation is a key factor in Enterprise Failure Prediction. A spike in latency at the time of a product release is not necessarily an issue, however, if it happens during low traffic, it likely means there is a significant issue. A rare log sequence can be considered normal in a controlled deployment but not during steady state operation, when it might point to a problem. Therefore, behavioural pattern modelling should capture release windows, topology and workload seasonality and maintenance state. If this context is missing, models can generate genuine operational variation as false alarms.

Figure 4 shows a view of failure escalation integrated. It illuminates the ways that weak deviations combine with each other in interaction, with persistence over time, and with the surrounding context to create operational risk. This is important because enterprise failure prediction can be less about an outlier point than a sequence of points. The figure illustrates a typical progression path in which weak deviations grow into failures as a result of their persistence, cross-signal agreement, topology exposure, and limited remediation capabilities. It reinforces the notion that failure of an enterprise should be modelled as episodes and escalation mechanism instead of isolated anomalies. Integrative interpretation of the result of cloud operations, storage prognostics, streaming anomaly detection and predictive-maintenance studies [6] – [17].

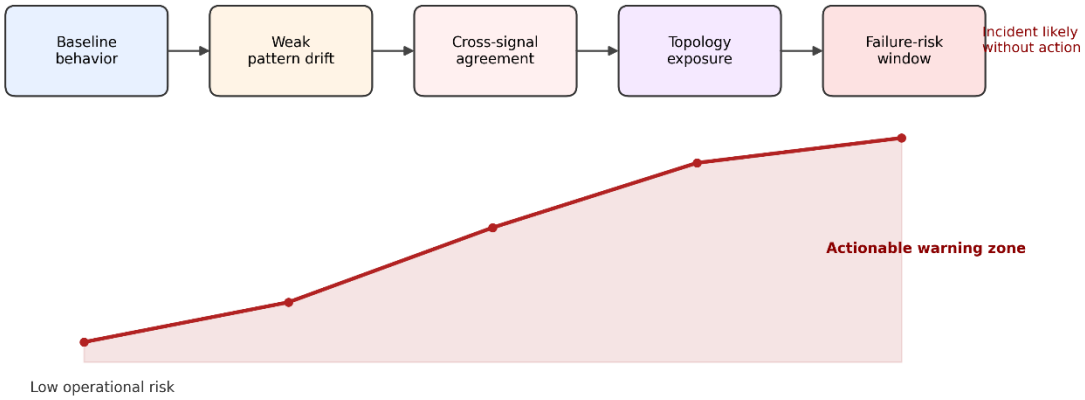

Figure 4: Behavioural Escalation from Deviation to Enterprise Failure Risk

There are two implications of the integrative figure. First, the features of escalation, e.g., persistence of anomalies, criticality of the entity, agreement of cross-modal, and dependency position should be modelled by enterprise failure prediction. Secondly, it is important to differentiate between early weak warnings and actionable warnings. Late detection can be operationally useless while early detection can be too noisy. Metrics typically provided in current studies are based on anomalies at a point level, while the aim of failure prediction is to provide a metric based on a set of episodes (snapshots), where the metric should give credit for the ability to detect a failure on time and it should penalize false alarms that would not occur.

One consistent theme in the reported studies is the need for the implementation of behavioural pattern modelling as a supplement to other measures. To develop intelligent failure prediction, it is necessary to map from behaviour to risk within enterprise constraints. Literature examined elements of this mapping: log and metric alignment [6], temporal adaptation [9], seasonal baseline modelling [10], deep subsequence detection [11], degradation-oriented prognostics [13] and deployment-

oriented predictive maintenance principles [15]. What is missing is an integrated assessment and modelling tool to relate these to incidents in the enterprise, value of intervention and system topology. This is why so many models look good in theory but are hard to get into place in large organizations.

V. FUTURE DIRECTIONS

Future work should focus on failure-oriented behavioural modelling, and not on anomaly scoring in general. The first need is to have development of incident-calibrated objectives that estimate probability, lead time, and severity of enterprise failure episodes. Such goals should clearly set out a difference between harmless novelty and behaviour that raises operational risk. This will need training data comprising of logs, metrics, traces, deployment data, incident tickets and remediation actions. Censored outcomes also must be handled carefully as it could be possible that a successful intervention could cause a failure label to not be applied.

The second priority is Topology-aware behavioural modelling. Failure is not due to local deviations, but to dependencies in enterprise systems. Context for behavioural deviations should be represented as service graphs, storage dependencies, network paths, and workload-routing policies. Current log–metric and time-series models can be enhanced by graph features and/or dependency features, but it is important to assess if topology helps to improve the quality of the warnings or to make models more complex. The literature on microservices suggests that operational dependency is a key issue for reliability risk [2] but there is little evidence from the journals of topology-integrated failure prediction.

The third priority relates to interpretability. There is deep anomaly detection review, in which the representation learning model complex deviations, which can also hide the importance of the alert [4],[17],[18]. Explanations are needed based on observed behaviour such as unusual event sequence, metric divergence, topology exposure, degradation slope or release-associated drift for enterprise operators. Future models need, therefore, to generate a compact causal or quasi-causal story, rather than scores. In order to achieve this goal, fault-injection studies can be helpful in providing controlled failure instances, but the faults inserted should also be checked against real failures that occur in production systems so as to avoid an artificial sense of security [19].

One of the fourth priorities is benchmark design. There are few public datasets available for enterprise failure prediction, with the few that are available often lacking logs, metrics, traces, topology, change events, incident outcomes, and remediation history in one coordinated timeline. Comparative statements are not always reliable in the absence of these benchmarks. Time-series forecasting and anomaly-detection approaches offer valuable baselines [10, 16, 20] but enterprise failure prediction benchmarks must incorporate lead-time windows, alert budgets, costs of false alarms, and effects of intervention after an alert. Operational metrics (mean warning time before incident, actionable precision, incident-reduction potential and software release stability) should be reported in future studies.

Last but not least, interdisciplinary integration of reliability engineering, machine learning, human factors and service management. Automated behavioural modelling with operator feedback, domain constraints, and adaptive threshold governance will be the most valuable future systems. It's not just a more complex learning architecture in the center that is the opportunity. The central opportunity is a technically correct, explainable, decision-calibrated, and change-resilient failure prediction.

VI. CONCLUSION

The field of intelligent failure prediction in enterprise systems via behavioural pattern modelling is an important yet underrepresented research area. The literature reviewed reveals that deviations in the sequences of events, in the metrics' evolution, in the condition of the components, in their workloads, or in the relationships between them tend to be structured and precede operational failures. Behavioural modelling is thus more appropriate than the use of isolated thresholds to monitor degradation and emerging instability.

Anomaly detection and failure prediction are distinct, yet related concepts; this is the central insight of the scholarly work presented here. Anomaly models detect deviations from learned behaviour, and failure prediction needs to relate the deviations with risk, time, system context, and operational action. Log–metric correlations, temporal forecasting, streaming anomaly detection, recurrent health-state modelling and prognostic deep learning are the most promising approaches found in the literature. None of these strands, however, is completely a solution to enterprise failure prediction by itself.

Persistent gaps include limited multimodal data integration, weak topology representation, unstable labels, limited explainability, little treatment of intervention-censored outcomes, and insufficient validation across production environments. Prospective advances require modelling that sees failures as unfolding behaviour events rather than abnormal points. The field is progressing towards risk-aware, context-aware, decision-calibrated prediction, but with a need for more empirical evidence before intelligent failure prediction can be a reliable part of enterprise operation.

VII. REFERENCES

[1] He, S., He, P., Chen, Z., Yang, T., Su, Y., & Lyu, M. R. (2021). A survey on automated log analysis for reliability engineering. *ACM Computing Surveys, 54*(6), Article 130.

[2] Soldani, J., Tamburri, D. A., & Van Den Heuvel, W.-J. (2018). The pains and gains of microservices: A systematic grey literature review. *Journal of Systems and Software, 146*, 215–232.

[3] Blázquez-García, A., Conde, A., Mori, U., & Lozano, J. A. (2021). A review on outlier/anomaly detection in time series data. *ACM Computing Surveys, 54*(3), Article 56.

[4] Pang, G., Shen, C., Cao, L., & Van Den Hengel, A. (2021). Deep learning for anomaly detection: A review. *ACM Computing Surveys, 54*(2), Article 38.

[5] Carvalho, T. P., Soares, F. A. A. M. N., Vita, R., Francisco, R. D. P., Basto, J. P., & Alcalá, S. G. S. (2019). A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering, 137*, Article 106024.

[6] Farshchi, M., Schneider, J.-G., Weber, I., & Grundy, J. (2018). Experience report: Anomaly detection of cloud application operations using log and cloud metric correlation analysis. *Journal of Systems and Software, 137*, 531–549.

[7] Susto, G. A., Schirru, A., Pampuri, S., McLoone, S., & Beghi, A. (2015). Machine learning for predictive maintenance: A multiple classifier approach. *IEEE Transactions on Industrial Informatics, 11*(3), 812–820.

[8] Xu, C., Wang, G., Liu, X., Guo, D., & Liu, T.-Y. (2016). Health status assessment and failure prediction for hard drives with recurrent neural networks. *IEEE Transactions on Computers, 65*(11), 3502–3508.

[9] Ahmad, S., Lavin, A., Purdy, S., & Agha, Z. (2017). Unsupervised real-time anomaly detection for streaming data. *Neurocomputing, 262*, 134–147.

[10] Taylor, S. J., & Letham, B. (2018). Forecasting at scale. *The American Statistician, 72*(1), 37–45.

[11] Munir, M., Siddiqui, S. A., Dengel, A., & Ahmed, S. (2019). DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access, 7*, 1991–2005.

[12] Zhao, R., Yan, R., Chen, Z., Mao, K., Wang, P., & Gao, R. X. (2019). Deep learning and its applications to machine health monitoring. *Mechanical Systems and Signal Processing, 115*, 213–237.

[13] Li, X., Ding, Q., & Sun, J.-Q. (2018). Remaining useful life estimation in prognostics using deep convolution neural networks. *Reliability Engineering & System Safety, 172*, 1–11.

[14] Lei, Y., Li, N., Guo, L., Li, N., Yan, T., & Lin, J. (2018). Machinery health prognostics: A systematic review from data acquisition to RUL prediction. *Mechanical Systems and Signal Processing, 104*, 799–834.

[15] Zhang, W., Yang, D., & Wang, H. (2019). Data-driven methods for predictive maintenance of industrial equipment: A survey. *IEEE Systems Journal, 13*(3), 2213–2227.

[16] Cook, A. A., Mısırlı, G., & Fan, Z. (2020). Anomaly detection for IoT time-series data: A survey. *IEEE Internet of Things Journal, 7*(7), 6481–6494.

[17] Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., & Müller, K.-R. (2021). A unifying review of deep and shallow anomaly detection. *Proceedings of the IEEE, 109*(5), 756–795.

[18] Chalapathy, R., & Chawla, S. (2019). Deep learning for anomaly detection: A survey. *WIREs Data Mining and Knowledge Discovery, 9*(3), Article e1316.

[19] Natella, R., Cotroneo, D., & Madeira, H. S. (2016). Assessing dependability with software fault injection: A survey. *ACM Computing Surveys, 48*(3), Article 44.

[20] Torres, J. F., Hadjout, D., Sebaa, A., Martínez-Álvarez, F., & Troncoso, A. (2021). Deep learning for time series forecasting: A survey. *Big Data, 9*(1), 3–21.