Street Art & Urban Creativity | Vol. 11, No. 7s, 2025 | ISSN 2183-9956
Street Art & Urban Creativity. Scientific Journal

# Self-Optimizing Industrial Intelligence

## A Multimodal AI Framework for Predictive Maintenance and U.S. Manufacturing Productivity

CHAITANYA NAGABHYRU[1]
[1]12-1006, RAHMATH NAGAR 2ND LINE, CHILAKALURIPET, GUNTUR PIN: 522616, ANDHRA PRADESH, INDIA
CHAITANYANAGABHYRU@GMAIL.COM

## Abstract

Unplanned equipment downtime erodes the productivity of U.S. factories, and the machine-learning models built to anticipate it often decay when equipment is upgraded or operating conditions change. This paper presents a multimodal predictive-maintenance framework that fuses machine sensor streams, equipment alarm logs, technician maintenance notes and production data, monitors each data type for drift, and adapts the model when drift is confirmed. A design layer for retrieval-based work-order assistance with expert validation is described but not evaluated. Because plant data cannot be shared, the framework is evaluated on a transparent synthetic fleet of 160 machines over 900 days, in which a retrofit of 40% of machines, a heavier workload and new production mix are introduced at day 450. Fusing all four data types raises average precision for seven-day failure prediction from 0.32 for sensors alone to 0.46 (late fusion) and 0.47 (early fusion) after the change. Drift monitors flag the production and sensor shifts within 2 and 9 days with no false alarms before the change, but the log shift is missed by distribution monitoring. Adaptation recovers part of the loss on retrofitted machines (average precision 0.39 for the static model, 0.44 with drift-triggered adaptation, 0.47 with quarterly retraining), though it does not beat simple periodic retraining. Alerts precede failures by 7.5 days on average. In a 510-day maintenance simulation, model-based policies reduce unplanned downtime by 33% to 41% with less than half as many interventions as a 90-day calendar schedule, but a 90-day calendar schedule has lower total cost under the base assumptions and at most tested cost settings, and no model-based policy was the cheapest option at any tested setting. All results are simulation-based.

**Keywords:** Predictive Maintenance, Multimodal Learning, Concept Drift, Adaptive Learning, Advanced Manufacturing, Fault Detection, Industrial Automation.

## 1. Introduction

Advanced manufacturing is among the critical and emerging technology areas that U.S. policy identifies as central to economic strength and supply-chain resilience [1]. In semiconductor fabrication, logistics hubs and discrete manufacturing alike, the productive capacity of a plant is set not only by how many machines it owns but by how reliably they run. A single unplanned stoppage of a processing tool or a conveyor can idle downstream stations, spoil work in progress and ripple through delivery schedules. Industry 4.0 architectures promise that connected machines and analytics will turn this exposure into foresight [2], and predictive maintenance, which schedules interventions according to condition rather than the calendar, is among the most cited applications [3], [5].

Predictive models, however, rarely stay accurate. A model trained on one operating regime can fail when a machine is retrofitted with new sensors or firmware, when a plant moves to a heavier product mix, or when a new failure mode appears. Reviews of the field note that most published work evaluates models on stationary benchmark data and pays little attention to what happens after deployment [3], [6]. A second difficulty is that the evidence about a machine is spread across data types that differ in form, frequency and reliability: continuous sensor streams, event-driven alarm logs, free-text technician notes written weeks apart, and production counts.

Multimodal learning offers ways to combine such sources [7], but each source can drift on its own schedule, so a fused model needs to know which of its inputs has changed.

This paper develops and evaluates a framework that addresses both difficulties. The objectives are: (i) to fuse sensor, log, text and production data into a single short-horizon failure predictor; (ii) to monitor each data type for drift and detect performance degradation; (iii) to adapt the model when drift is confirmed, without switching to a worse model; (iv) to translate predictions into maintenance decisions and measure their consequences for downtime, cost and throughput; and (v) to describe how risk alerts can feed an expert-validated work-order assistant, along the lines of retrieval-augmented generation [18]. Performance is measured by unplanned downtime, maintenance cost, fault-detection lead time and production throughput.

The topic is one of national interest: more productive domestic manufacturing, more resilient supply chains and less downtime. Realistic labeled failure histories from factories are proprietary, so every experiment uses a synthetic fleet whose construction is described in Section V. The results should be read as properties of the method on that benchmark, and several of them are not flattering to the proposed adaptation procedure, which is reported as it performed. The paper is organized as follows: Section II reviews the literature, Section III positions related work, Section IV presents the methodology, Section V the experimental setup, Section VI the results, Section VII the discussion and limitations, and Section VIII the conclusions.

## 2. LITERATURE REVIEW

Condition-based maintenance has a long history in mechanical engineering, where diagnostics identify a developing fault and prognostics estimate time to failure from vibration, temperature, acoustic and similar signals [3]. Surveys of data-driven methods catalogue statistical, classical machine-learning and deep-learning approaches, and conclude that the choice of method matters less than data quality, label availability and the match between the evaluation protocol and the deployed setting [4], [5], [21]. Deep networks applied to raw signals can learn features automatically but need large labeled sets, whereas tree ensembles on engineered window statistics are competitive on tabular summaries and are easier to audit [6], [9].

Fusing heterogeneous data is the subject of the multimodal learning literature, which distinguishes early fusion, where features of all modalities are combined before learning, from late fusion, where each modality is modeled separately and the outputs are combined [7]. Stacked generalization, which trains a second-level model on the outputs of first-level models, is a standard form of late fusion [8]. Late fusion tolerates missing modalities and allows modality-specific maintenance, while early fusion can capture interactions across modalities. Text in maintenance records can be represented by weighted term counts [10] and compressed by low-rank factorization [11], or by learned embeddings in modern systems.

Concept drift, the change over time of the relationship between inputs and the target, has been surveyed extensively. Reviews distinguish sudden, gradual and recurring drift and separate approaches that detect drift from those that adapt blindly through windowing or ensembles [12], [13]. Detection methods include sequential tests on error streams, such as adaptive windowing [14] and cumulative-sum charts [15], and tests that compare the distribution of inputs with a reference, which can work without labels [16]. The population stability index, widely used in risk modeling, summarizes the shift in a feature's distribution across bins [17]. Label delay is a defining constraint in maintenance: whether a machine fails within seven days is known only after seven days, so performance-based detection lags input-based detection.

Retrieval-augmented generation couples a language model with a knowledge base of documents indexed by embeddings, so that generated text is grounded in retrieved evidence [18]. In maintenance, the natural documents are historical work orders, technician notes and equipment manuals, and the outputs are work-order drafts with instructions and spare-parts information, subject to review by subject-matter experts. Evaluating prediction systems on imbalanced events calls for precision-recall measures, since the area under the ROC curve can look high when events are rare [19].

## 3. RELATED WORK

Three groups of studies are closest to this work. The first concerns multimodal predictive maintenance, in which authors combine vibration, process variables and sometimes text from

maintenance logs. These studies generally show that adding modalities improves detection on a fixed dataset, as the ablation in Section VI also finds, but they typically evaluate on a random split of one period and say little about behavior after a change in equipment [4], [5]. The second concerns drift handling in industrial streams, where windowed retraining and drift detectors are standard [12], [14]. The difficulty reported repeatedly is that detectors based on a single summary signal either miss shifts that are confined to a subset of machines or raise false alarms under normal variation.

The third group comprises retrieval-based assistants for operations and maintenance, which index work orders and manuals in a vector store and generate recommended actions [18]. The design in this paper follows that pattern: structured sources (sensors, historians, control systems) and unstructured sources (past work orders, technician notes, manuals) are encoded by embedding models into a knowledge base, a retrieval-augmented generator drafts the work order with a template, spare-parts information and repair instructions, and a subject-matter expert validates the draft, with the expert's feedback available for later learning. What this paper adds is the upstream predictor with drift handling that supplies the trigger and the risk context. The assistant itself is described as a design, and no language-model component is evaluated here.

**TABLE 1. Positioning of the proposed framework relative to common practice**

| Capability | Common practice | Treatment in this framework |
|---|---|---|
| Data types | Sensors only, or one added source | Sensors, alarm logs, technician notes and production data |
| Missing or weak modalities | Imputation of whole records | Modality-wise models with late stacking, or native handling of missing values |
| Change in equipment | Periodic retraining or none | Per-modality drift monitors and validated, targeted retraining |
| Evaluation | Random split of one period | Chronological split with an engineered drift event and per-subgroup results |
| Decisions | Alert accuracy only | Downtime, cost and throughput in a budget-limited maintenance simulation |

The contribution is the integration of these elements and their joint evaluation, including the results that do not favor the proposed components. No individual technique is claimed to be new. As in most of this literature, the lack of open fleet-scale failure histories limits what can be claimed about deployment.

## 4. METHODOLOGY

### *Framework Overview*

Figure 1 summarizes the framework. Three groups of sources feed it: structured data (sensor streams, historian and control-system tags, production counts), semi-structured data (equipment alarm and event logs) and unstructured text (technician notes, old work orders, manuals). Each group has an encoder that turns a recent window of data into features. The encoded modalities are fused to produce, for each machine each day, a risk score for failure within the next seven days. A drift monitor watches each modality and the model's recent performance. When drift is confirmed, an adaptive learner retrains the affected modality, checks the new model on recent labeled data, and switches to it only if it is better. Risk scores and lead-time alerts feed maintenance planning, and a work-order assistant, described below, turns alerts into draft work orders for expert validation.

.




***Figure 1: Architecture of the proposed multimodal predictive-maintenance framework with drift monitoring, adaptive learning and an expert-validated work-order assistant.***

## *Encoders for Each Data Type*

The sensor encoder describes each of four signals (vibration, temperature, pressure and power) over the latest seven days by its average, variability and short-term slope, and by how far the recent average sits above the machine's own earlier thirty-day baseline, both as a ratio and in baseline standard deviations. Using the machine's own baseline makes the features partly robust to changes in sensor calibration, though not to changes in what the sensor signals mean. The log encoder counts each of twelve alarm codes over the last three and the last fourteen days and totals all alarms over the last seven days. The production encoder compares the recent seven-day average of units produced, yield and cycle time with the earlier baseline and adds the variability of daily output. The text encoder converts the most recent technician note into weighted term counts [10], reduces them to six dimensions by low-rank factorization [11], and adds the age of the note. A context group, shared by all models, gives the machine family and the days since its last service. Table 2 lists the groups.

**TABLE 2. Data modalities and encoded features**

| Modality | Raw data | Encoded features | Available |
|---|---|---|---|
| Sensors | Vibration, temperature, pressure, power, daily | Recent mean, variability, slope, ratio to own baseline, baseline z-score (20) | Daily, with 3% dropout and sensor-free periods during repairs |
| Equipment logs | Twelve alarm codes | Counts over 3 and 14 days, 7-day total (25) | Daily |
| Technician notes | Short free-text notes | Six text components and note age (7) | About every 4 weeks, plus after service |
| Production data | Units, yield, cycle time | Relative change in output, yield and cycle time, output variability (4) | Daily |
| Context | Machine family, service history | Family code, days since service (2) | Daily |

## *Fusion Strategies*

Two fusion designs are compared. In late fusion, a gradient-boosted tree classifier [9] is trained separately for each modality, always with the context features. The four resulting probabilities are converted to log-odds and combined by a logistic regression trained on a separate, later period, in the manner of stacked generalization [8]. In early fusion, one boosted classifier is trained on all features together. Both handle missing values natively. The target is

whether the machine fails within the next seven days, and rows from machines that are down on a given day are excluded.

### *Drift Monitoring*

Two kinds of monitor run weekly. The first compares the distribution of each modality’s features in the last fourteen days with a reference sample from the training period, using the population stability index [17] averaged over the modality’s features. Its alarm threshold is not chosen by judgment: it is one and a half times the largest value observed on weekly windows during the validation period, a period without drift, so that a false alarm under normal variation is unlikely by construction. The second monitor tracks the average squared error of the fused probabilities on the most recent twenty-one days of matured labels (those at least seven days old) and signals when it exceeds one and a half times its validation value. Input monitoring is fast and works without labels; performance monitoring is slower because of label delay but catches shifts that the input monitors miss.

### *Drift-Triggered Adaptation*

When a modality has been flagged for at least three weeks, or the performance monitor has fired, the adaptive learner retrains the affected modality model. It uses the most recent six months of labeled data, giving double weight to the latest three and a half months, and refits the text encoder’s vocabulary when the text modality is retrained. The candidate is trained on data up to forty-five days before the present and is compared with the current model on the last forty-five days of matured labels. It replaces the current model only if its logarithmic loss on that held-out period is at least two percent lower. After any accepted change, the stacking model and the alert threshold are refit on the same held-out period, so they are always fit on data that no component has been trained on, and the drift reference for the changed modality is reset. A fresh attempt for the same modality is not made within four weeks. In the early-fusion variant, the single model is retrained in the same way whenever a modality-level change is accepted, and replaced under the same two percent rule. Each model version is used from the day after the update.

Two comparison strategies are included. Periodic retraining replaces all four modality models every ninety days (quarterly) or every thirty days (monthly), using the most recent labeled data and a held-out final sixty days for stacking, whether or not drift has occurred. A static model is never updated. Alert thresholds are set so that about two percent of machine-days on the calibration data raise an alert.

### *Maintenance Decisions*

Predictions are turned into decisions by a simple, budget-limited rule. Each day, among machines not serviced in the last fourteen days whose risk exceeds the alert threshold, the crew performs preventive maintenance on up to four machines in order of risk. Comparison policies are no preventive work (reactive), calendar maintenance every 90 or 180 days, and an oracle that services machines whose true wear exceeds eighty percent of life. Preventive maintenance removes about two thirds of accumulated wear, whereas a repair after a failure restores the machine almost fully. Performance is measured over a 510-day period by unplanned downtime, number of failures and preventive actions, production throughput and total cost.

### *Work-Order Assistant (Design)*

The final layer, not evaluated in this paper, follows the design of retrieval-based work-order generation. Structured streams and unstructured documents, including past work orders, technician notes and manufacturer manuals, are embedded by separate embedding models into a vector knowledge base. When an alert is raised, a retrieval-augmented generator retrieves similar past cases, relevant manual sections and spare-parts information, and drafts a work order with a template, a parts list and repair instructions. A subject-matter expert reviews and edits the draft before release, and the decisions are stored so that the knowledge base and the generator can improve. Table 3 lists the business roles that interact with the system and what each receives.

**TABLE 3. Roles and outputs of the framework**

| Role | Receives | Decision supported |
|---|---|---|
| Asset manager | Fleet risk ranking, trend of unplanned downtime, cost scenarios | Maintenance budget and intervals |
| Operations and engineering | Machine-level alerts, contributing modalities, draft work orders | Whether and when to take a machine offline |
| Data scientist | Drift reports, retraining history, model comparison | Model upkeep and validation |
| Procurement | Spare-parts lists attached to draft work orders | Parts ordering and stock |
| Control-room operator | Short-horizon alerts with lead time | Scheduling around expected failures |

## 5. EXPERIMENTAL SETUP

### *Synthetic Fleet*

The benchmark simulates 160 machines of four families (40 each) for 900 days. Each machine has four wear components (bearing, seal, thermal and electrical) that accumulate at rates depending on workload, with random day-to-day variation and a component life of roughly 220 to 340 days. A failure occurs when any component reaches its life, or at random through a sudden fault with a hazard of one in 700 machine-days that has no warning signal. After a failure the machine is down for two to five days and then restored almost fully. In addition to these, machines receive no preventive work during the history that precedes deployment.

Observations follow from the wear state. Vibration rises with bearing wear, pressure falls with seal wear, temperature rises with thermal wear, and power rises with electrical wear. Alarm codes occur at a low base rate that rises with the wear of the component the code belongs to. Production falls slightly with wear. Technicians write a short note about every four weeks and after service; if a component is more than 40% worn, the note describes its symptom three times out of four, otherwise it says the machine is normal. Sensors have noise, occasional dropout, and are absent during repairs. Figure 2(a) shows the vibration and temperature of one machine, with failures and the change point.

At day 450, three changes occur together. First, 40% of machines are upgraded: sensor calibration shifts, the vibration and pressure signatures of bearing and seal wear weaken, the electrical fault begins to show in temperature and no longer in power, alarm codes are renumbered by the new firmware, and technicians' notes use new terminology. Second, every machine runs a heavier workload that speeds wear by 20%. Third, the product mix changes so that output per machine and cycle time rise. These changes are of the three types usually discussed in the drift literature: shifts in the inputs, in the relationship between inputs and failure, and in operating conditions. The severity is a design choice, and the results depend on it. Failures per month increase after the change (Figure 2(b)).

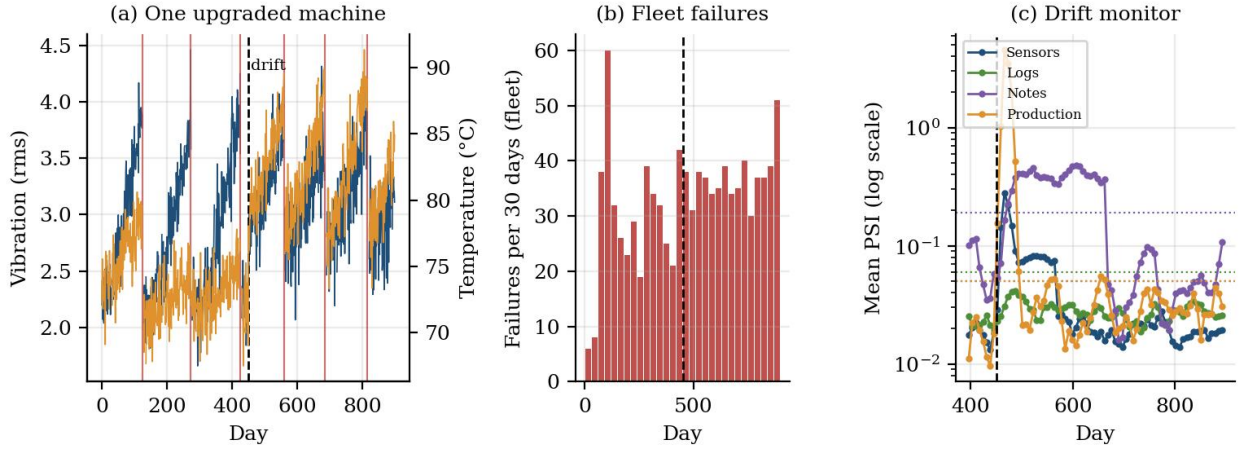



***Figure 2: (a) vibration and temperature for one upgraded machine, with failures (red lines) and the change point; (b) fleet failures per 30 days; (c) population stability index by modality in one simulation, with alarm thresholds (dotted)***

### *Protocol*

Days 36 to 322 form the training period (the first 36 days supply feature history, and the last labels in this period are known by day 329). Days 330 to 382 form the calibration period for stacking and alert thresholds, and 330 to 389 also provide the drift-monitor thresholds. The deployment period runs from day 390 to day 899, of which days 390 to 449 precede the change and 450 to 899 follow it. Failure labels for day d are known only after day d plus seven, which governs what any adaptive procedure may use. Three independent simulations were run, each with a new fleet, new noise and new training, and results are reported as mean ± standard deviation across simulations. Over the deployment period there are on average 593 failures, of which 534 follow the change. Boosted classifiers use 80 iterations, a learning rate of 0.1, at most 15 leaves and at least 40 examples per leaf, with scikit-learn [20].

**TABLE 4. Principal experimental settings**

| Component | Setting |
|---|---|
| Fleet | 160 machines in 4 families; 900 days; 4 wear components per machine |
| Change event | Day 450: 40% of machines upgraded, 20% heavier workload, new production mix |
| Prediction target | Failure within the next 7 days |
| Alert threshold | 98th percentile of calibration scores (about 2% of machine-days) |
| Drift monitors | Weekly population stability index per modality; error monitor on matured labels |
| Adaptation | 6-month window, held-out validation of 45 days, 2% improvement required |
| Maintenance crew | Up to 4 preventive actions per day, 14-day cooldown |
| Base costs | Repair $60,000; preventive action $30,000; lost output valued at $80 per unit |
| Repetitions | 3 simulations |

Cost assumptions matter greatly to maintenance conclusions and are not drawn from any plant. Output is valued at 80 dollars per unit, so a machine-day of production is worth about 8,000 to 13,000 dollars depending on the machine family. A repair costs 60,000 dollars in parts and labor and also loses the output of the days the machine is down. A preventive action costs 30,000 dollars and loses one day of output. the sensitivity analysis in Section VI varies these assumptions.

## 6. RESULTS

### *Effect of Fusing Data Types*

Table 5 reports failure prediction across the deployment period. With sensors alone, average precision is 0.45 ± 0.03 before the change and 0.32 ± 0.01 after it. Adding the alarm log lifts post-change average precision to 0.45 ± 0.02, and adding technician notes to 0.45 ± 0.02. Late fusion of all four modalities reaches 0.46 ± 0.03 and early fusion 0.47 ± 0.02. The log is the single most valuable addition, because alarms respond to every wear component, and unlike a sensor reading it is largely independent of calibration. Notes and production data add smaller increments, in line with their sparse and indirect nature. Early fusion is slightly better than late fusion after the change (0.47 against 0.46), although before the change late fusion is better (0.53 against 0.50), so the two are indistinguishable given the variation between simulations. ROC areas are shown for completeness, but average precision is emphasized because failures are rare [19].

**TABLE 5. Failure prediction and alerting in the deployment period (mean ± sd over 3 simulations)**

| Strategy | AP before change | AP after change | AUC after change | Failures alerted (%) | Mean lead (days) | False alert days per machine-year |
|---|---|---|---|---|---|---|
| Sensor only | 0.45 ± 0.03 | 0.32 ± 0.01 | 0.837 ± 0.014 | 37.7 ± 3.3 | 9.1 ± 0.2 | 2.3 ± 0.5 |
| Sensors + logs | 0.50 ± 0.02 | 0.45 ± 0.02 | 0.872 ± 0.013 | 36.6 ± 5.1 | 7.5 ± 0.0 | 0.8 ± 0.1 |
| Sensors + logs + notes | 0.53 ± 0.02 | 0.45 ± 0.02 | 0.870 ± 0.013 | 42.4 ± 3.0 | 7.5 ± 0.1 | 1.0 ± 0.2 |
| Four modalities, late fusion (static) | 0.53 ± 0.02 | 0.46 ± 0.03 | 0.873 ± 0.017 | 43.3 ± 4.9 | 7.4 ± 0.2 | 0.9 ± 0.2 |
| Four modalities, early fusion (static) | 0.50 ± 0.01 | 0.47 ± 0.02 | 0.879 ± 0.009 | 43.7 ± 3.5 | 7.5 ± 0.2 | 0.9 ± 0.1 |
| Periodic retrain, quarterly | 0.53 ± 0.02 | 0.48 ± 0.01 | 0.880 ± 0.009 | 48.9 ± 0.9 | 7.2 ± 0.2 | 1.0 ± 0.2 |
| Periodic retrain, monthly | 0.50 ± 0.02 | 0.47 ± 0.02 | 0.879 ± 0.009 | 50.8 ± 0.4 | 7.3 ± 0.1 | 1.2 ± 0.2 |
| Drift-triggered, late fusion (proposed) | 0.53 ± 0.02 | 0.48 ± 0.01 | 0.881 ± 0.009 | 44.3 ± 0.8 | 7.1 ± 0.3 | 0.8 ± 0.2 |
| Drift-triggered, early fusion (proposed) | 0.50 ± 0.01 | 0.47 ± 0.01 | 0.884 ± 0.009 | 48.9 ± 0.3 | 7.3 ± 0.2 | 1.0 ± 0.1 |

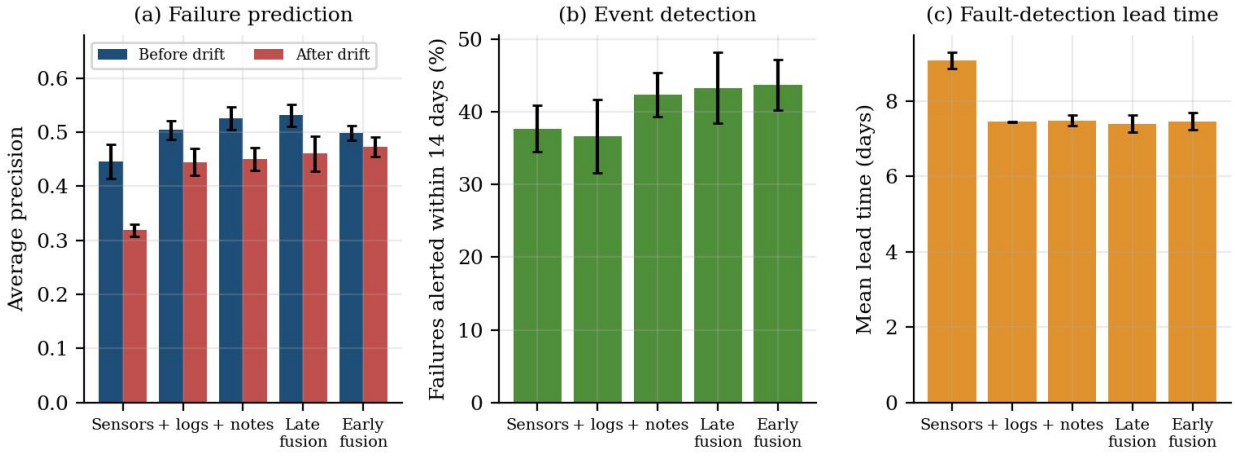



***Figure 3: Effect of adding data types on (a) average precision before and after the change, (b) share of failures alerted within 14 days and (c) mean lead time***

The effect of the change on the static models is moderate. Late fusion falls from an average precision of 0.53 before the change to 0.46 after it, and its ROC area from 0.897 to 0.873. The loss is concentrated in upgraded machines, where average precision after the change is 0.39, compared with 0.46 for all machines. This is partly because machines that were not upgraded carry on as before, and partly because relative-to-baseline features and the alarm log retain some value after the upgrade.

## *Drift Detection*

Table 6 summarizes the drift monitors. The production modality is flagged first, within 2 days of the change in every simulation, because its shift is large and affects every machine. The sensor modality is flagged after 9 days, which is the time needed for a fourteen-day window to contain enough post-change observations from the upgraded machines. The text modality is flagged after 23 to 30 days, because notes are written only every four weeks. No false alarm was raised by any input monitor before the change, as intended by the threshold design. The log modality was never flagged. Renumbering the alarm codes in upgraded machines leaves the marginal distribution of each count almost unchanged, because the codes have similar base rates, so an average of per-feature shifts is blind to it, even though the relationship between codes and wear changes. The performance monitor, which relies on matured labels, fired 30 to 79 days after the change and not before it. Input monitoring is therefore faster, whereas the performance monitor is needed to cover shifts that distributions cannot reveal.

**TABLE 6. Drift detection: days from the change to the first alarm, and alarms before the change**

| Monitor | Delay, simulation 1 | Delay, simulation 2 | Delay, simulation 3 | False alarms before change |
|---|---|---|---|---|
| Production data | 2 | 2 | 2 | 0 |
| Sensors | 9 | 9 | 9 | 0 |
| Technician notes | 23 | 23 | 30 | 0 |
| Alarm logs | not detected | not detected | not detected | 0 |
| Error on matured labels | 79 | 58 | 30 | 0 |

## *Adaptive Learning*

Detection led to retraining slowly. Drift-triggered retraining required a flag to persist for three weeks, a candidate to be trained and a held-out comparison to be passed. In the three simulations, 14, 24, 19 candidate retrainings were attempted and 2, 3, 5 were accepted, and the first accepted retraining occurred on days 564, 669, 592, that is 114, 219, 142 days after the change. Many candidates were rejected, since they did not improve the held-out loss by two percent.

Figure 4 and Table 5 show the effect on predictive quality. On the upgraded machines, which are the machines whose behavior changed, average precision after the change is 0.39 ± 0.04 for the static late-fusion model, 0.42 ± 0.01 for drift-triggered late fusion, 0.44 ± 0.02 for drift-triggered early fusion, 0.47 ± 0.01 for quarterly retraining and 0.45 ± 0.01 for monthly retraining. Adaptation therefore recovers part of the loss, and the drift-triggered variants lie between the static model and periodic retraining. In the rolling view for one simulation (Figure 4(a)), all strategies degrade together immediately after the change, since no strategy has post-change labels at that time, and the strategies separate only after one to four months. Late in the deployment period the static model declines again while the adapted models hold, which suggests that the post-change regime was not fully stable. Across all machines, the differences in the ROC area after the change among the adaptive and periodic strategies are within about 0.006. The honest conclusion is that drift-triggered adaptation helps relative to never updating, but in this benchmark it does not outperform the simple schedule of periodic retraining, and the fraction of the drift problem it solves is limited by the delay in labels and by its conservative validation.

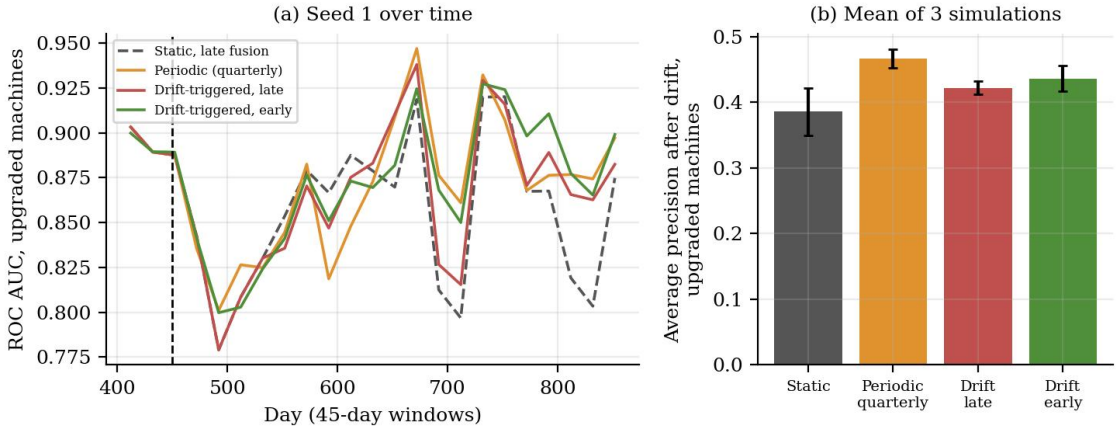



***Figure 4: (a) ROC area over time on upgraded machines for one simulation, with the change point (dashed); (b) average precision after the change on upgraded machines***

## *Fault-Detection Lead Time*

At the calibrated operating point, about 43% of failures receive an alert in the fourteen days before they occur, using multimodal late fusion, 44% using early fusion and 49% with quarterly retraining, against 38% with sensors alone. The share is limited in part by sudden faults without warning, which no model can detect, and in part by a conservative operating point of about two percent of machine-days. Among failures that are alerted, the first alert comes on average 7.5 days before the failure for early fusion and 7.4 days for late fusion. Sensors alone give a longer mean lead of 9.1 days, but for fewer failures and with the fourteen-day window truncating many long leads; Figure 5 shows that its distribution has a pile-up at the full fourteen days, which indicates alerts that begin long before the failure and may include persistent alerts on slowly

worn machines. The false-alert burden is about 0.9 to 0.9 alert days per machine-year outside pre-failure windows.

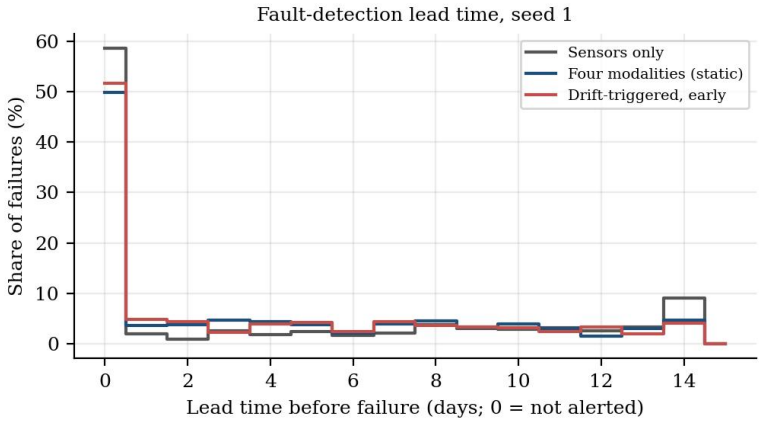



***Figure 5: Distribution of the lead time before failure for three strategies in one simulation (0 days means the failure was not alerted)***

## *Maintenance Outcomes*

Table 7 and Figure 6 report the maintenance simulation. With no preventive work, the 160 machines experience 593 failures and 2655 machine-days of unplanned downtime in 510 days, at a total cost of $104.3 million, of which most is repair and lost production. Model-based policies reduce failures to between 350 (early fusion with drift-triggered adaptation) and 397 (late fusion) and unplanned downtime by 34% to 41%, using 278 to 359 preventive actions. The differences among static, periodic and drift-triggered versions of the same model are small compared with variation between simulations. A calendar schedule of 90 days reduces unplanned downtime by 67% but needs 791 preventive actions, more than twice as many. The oracle, which knows true wear, reduces unplanned downtime by 82%.

Total cost under the base assumptions does not favor model-based policies. Calendar maintenance every 90 days costs $91.7 million, the drift-triggered early-fusion policy $99.9 million and the reactive baseline $104.3 million, so that the model-based policies save about 4% against reactive repair and cost 9% more than the 90-day calendar. Throughput tells a similar story. The 90-day calendar and the oracle raise throughput by about 1.7%, whereas model-based policies leave throughput within 0.1% of the reactive case, in spite of the downtime they avoid. The reason is the way the simulation treats wear. A repair after a failure almost fully restores a machine, whereas a preventive action removes only about two thirds of the accumulated wear, and production declines with wear. Model-based policies avoid some failures but leave machines more worn on average, and the lost output from higher wear offsets the output saved from shorter downtime. Frequent calendar maintenance keeps wear low throughout, which raises throughput.

**TABLE 7. Maintenance simulation over 510 days for 160 machines (mean ± sd over 3 simulations; base costs)**

| Policy | Failures | Unplanned downtime (machine-days) | Preventive actions | Throughput (million units) | Total cost ($M) |
|---|---|---|---|---|---|
| Reactive (run to failure) | 593 ± 21 | 2655 ± 93 | 0 ± 0 | 9.305 ± 0.005 | 104.3 ± 1.7 |
| Time-based (every 180 days) | 532 ± 26 | 2358 ± 128 | 91 ± 8 | 9.314 ± 0.005 | 102.6 ± 1.8 |
| Time-based (every 90 days) | 197 ± 26 | 863 ± 110 | 791 ± 13 | 9.462 ± 0.008 | 91.7 ± 1.8 |
| Static multimodal model | 402 ± 8 | 1778 ± 34 | 279 ± 23 | 9.312 ± 0.006 | 100.7 ± 1.5 |
| Static early-fusion model | 376 ± 10 | 1663 ± 45 | 316 ± 43 | 9.313 ± 0.002 | 100.1 ± 0.7 |
| Periodic retrain, quarterly | 376 ± 6 | 1681 ± 47 | 313 ± 21 | 9.313 ± 0.006 | 100.1 ± 1.3 |
| Drift-triggered, late fusion (proposed) | 397 ± 20 | 1756 ± 99 | 278 ± 6 | 9.312 ± 0.006 | 100.4 ± 1.4 |
| Drift-triggered, early fusion (proposed) | 350 ± 10 | 1567 ± 50 | 359 ± 25 | 9.313 ± 0.008 | 99.9 ± 1.3 |
| Oracle (true wear) | 109 ± 15 | 475 ± 62 | 885 ± 7 | 9.460 ± 0.005 | 89.4 ± 1.4 |

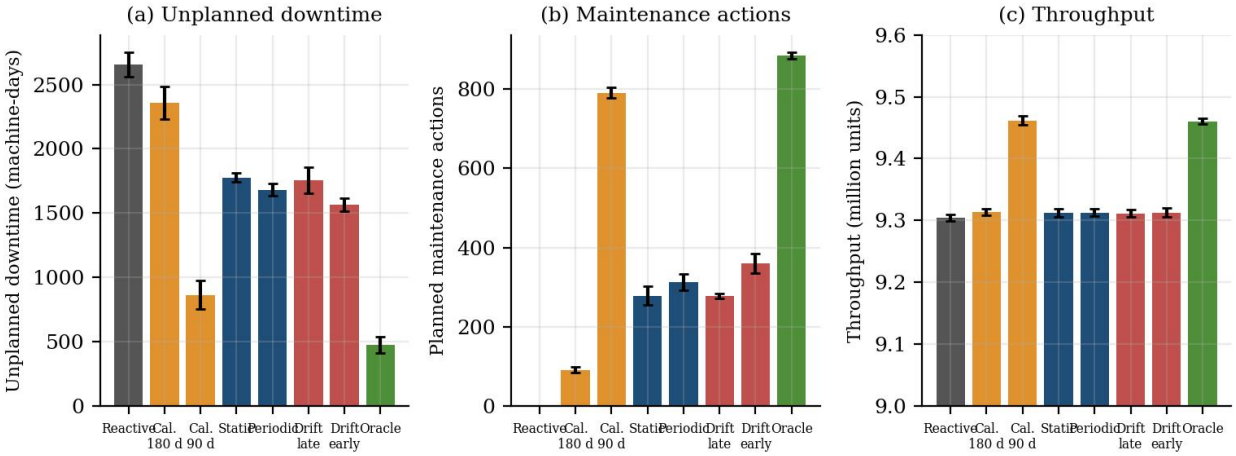



***Figure 6: Policy outcomes: (a) unplanned downtime, (b) number of preventive actions and (c) throughput over 510 days***

## *Sensitivity to Cost Assumptions*

Because the conclusion about cost depends on how expensive preventive actions are relative to failures, Table 8 and Figure 7 vary the cost of a preventive action from $10,000 to $60,000 and the cost of a repair from $60,000 to $120,000, recomputing total cost from the same simulated counts. When a preventive action costs $10,000, the drift-triggered early-fusion policy costs $92.7 million against $75.9 million for the 90-day calendar, so the calendar is far cheaper because its extra actions are inexpensive. The gap narrows as actions become costlier. At $45,000 the calendar still costs $103.6 million against $105.2 million for the model-based policy, and only at $60,000 per action is the model-based policy cheaper than the calendar ($110.6 million against $115.4 million), at which point every preventive policy, including the oracle ($115.9 million), costs more than running to failure ($104.3 million), since an action then costs as much as a repair. With a repair cost of $120,000 and an action cost of $30,000, the calendar costs $103.5 million, the model-based policy $120.8 million and the reactive policy $139.9 million. The model-based policy was therefore not the lowest-cost option at any tested setting. Its advantage lies elsewhere: it avoids a large share of unplanned downtime with less than half as many interventions as the calendar, which matters when maintenance crews, tool availability or process risk limit the number of preventive actions that can be performed.

**TABLE 8. Total cost ($M) over 510 days at different preventive-action costs (repair cost $60,000; mean over 3 simulations)**

| Policy | Action $10k | Action $20k | Action $30k | Action $45k | Action $60k |
|---|---|---|---|---|---|
| Reactive (run to failure) | 104.3 | 104.3 | 104.3 | 104.3 | 104.3 |
| Time-based (every 180 days) | 100.8 | 101.7 | 102.6 | 104.0 | 105.4 |
| Time-based (every 90 days) | 75.9 | 83.8 | 91.7 | 103.6 | 115.4 |
| Static multimodal model | 95.1 | 97.9 | 100.7 | 104.8 | 109.0 |
| Periodic retrain, quarterly | 93.8 | 96.9 | 100.1 | 104.8 | 109.5 |
| Drift-triggered, late fusion | 94.8 | 97.6 | 100.4 | 104.5 | 108.7 |
| Drift-triggered, early fusion | 92.7 | 96.3 | 99.9 | 105.2 | 110.6 |
| Oracle (true wear) | 71.7 | 80.5 | 89.4 | 102.6 | 115.9 |

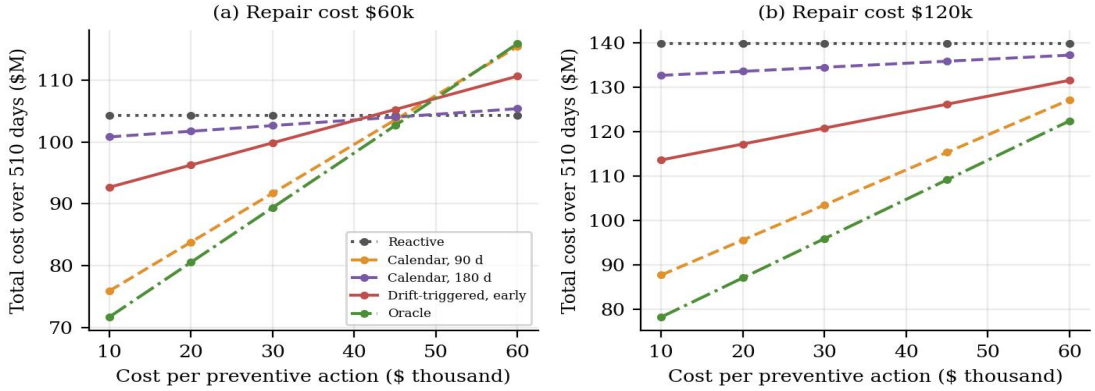



***Figure 7: Total cost versus cost of a preventive action for repair costs of (a) $60,000 and (b) $120,000***

## 7. DISCUSSION

Four points stand out. First, multimodal fusion helps, and the benefit comes from the data type that is most informative and least dependent on calibration. Alarm logs gave the largest gain over sensors alone. Text notes helped less in this benchmark, partly by construction, since notes are rare and sometimes uninformative, and partly because the benchmark's new terminology after the upgrade removes their value until the text encoder is refit. In a real plant, notes may carry more information than the simulation assumes, for example about parts and procedures, and the retrieval-based assistant described in Section IV uses them for a different purpose than risk scoring.

Second, monitoring each modality separately is informative, but distribution tests alone are not sufficient. The monitors found the production, sensor and text shifts quickly or within weeks and gave no false alarms in the validation protocol, but they missed the alarm-code renumbering, a change in the meaning of the data without a change in marginal distributions. Performance monitoring on delayed labels did detect it, at a cost of one to three months. A practical system should combine input monitors, a performance monitor, and, where possible, change records from the plant, such as firmware updates and retrofits, which here would have identified the affected machines immediately.

Third, the drift-triggered adaptation did not beat periodic retraining. It improved over a static model on retrofitted machines, but the combination of a three-week persistence rule, a forty-five-day validation period and a two percent improvement requirement delayed the first accepted update by several months after the change. The conservative validation prevents harmful updates, which is valuable, but here it also postponed useful ones. Looser rules or a quicker first retrain, with rollback if performance falls, might perform better, and a hybrid that combines periodic retraining with drift-triggered immediate updates is a natural design. The claim that drift-triggered adaptation is superior to periodic retraining is therefore not supported by this benchmark, though the drift monitors provide diagnostic value that a calendar schedule does not, including identifying which data type has changed.

Fourth, prediction quality and economic value are different things. Even with fairly accurate alerts, model-based maintenance saved only a modest share of cost under the base assumptions and lost to a 90-day calendar, because the benchmark's wear process is regular enough that calendar maintenance prevents most failures, and because partial preventive maintenance leaves wear that reduces output. The result should not be generalized to plants where failures are largely random, where preventive actions are expensive or invasive, or where over-maintenance itself causes faults, conditions common in semiconductor manufacturing and other complex process equipment. The sensitivity analysis shows the ranking depends on the ratio of action cost to failure cost. Plants should therefore estimate this ratio before investing in prediction, and the framework is most useful when it can be paired with decisions that are cheap to take, such as inspection or rescheduling, and not only full maintenance.

The limitations are considerable. The fleet is synthetic and the generating process, including the form and severity of the drift, is an assumption. Only three simulations were run, and the differences among adaptive and periodic strategies are within the variation between them. The scale and complexity of real semiconductor tools, with hundreds of correlated sensors, is absent. Maintenance policy results use models trained on a reference history without interventions,

while a real deployment would train on data affected by its own decisions. Failures are limited to four wear processes and a sudden fault, and the repair and restoration model is simple. The work-order assistant is a design that was not built or evaluated, and issues of retrieval quality, hallucination, and expert workload would need their own studies. Future work should validate on real fleet data, evaluate hybrid adaptation rules, include costs for false alarms and inspection, and test the assistant with maintenance engineers.

## 8. CONCLUSION

This paper presented a multimodal predictive-maintenance framework that fuses sensor, log, text and production data, monitors each data type for drift, adapts the model when drift is confirmed, and links predictions to maintenance decisions and to an expert-validated work-order assistant. On a synthetic fleet with a mid-period retrofit and change in workload, fusing all four modalities raised the average precision of seven-day failure prediction from 0.32 to 0.47 after the change. Drift monitors detected the production and sensor shifts within 2 and 9 days without false alarms but missed the log shift, and drift-triggered adaptation recovered part of the loss on retrofitted machines without surpassing simple periodic retraining. Alerts preceded failures by about 7 days. In a budget-limited maintenance simulation, model-based policies reduced unplanned downtime by up to 41% with far fewer interventions than a calendar schedule, but a 90-day calendar had lower total cost under the base assumptions, and no model-based policy was the cheapest option at any tested cost setting.

These conclusions describe a controlled benchmark and not field performance. The next steps are validation on real plant data, evaluation of faster adaptation rules, and a study of the work-order assistant with maintenance engineers, so that the benefits of adaptive industrial intelligence for U.S. manufacturing productivity are established on operational evidence.

## References


1. A Mahendra Vardhan and S. Sridhar, "Determining False Positive Analysis of Software Vulnerabilities with Predefined Scan Rules using Random Forest Classifier and Decision Tree Technique," 2022 4th International Conference on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida, India, 2022, pp. 622-625, DOI: https://doi.org/10.1109/ICAC3N56670.2022.10074458
2. Dip Bharatbhai Pate, Dynamic vs. Static visualizations: Understanding their use cases in big data analysis, International Journal of Science and Research Archive, 2022, 05(02), 392-395., DOI https://doi.org/10.30574/ijsra.2022.5.2.0077
3. Dip Bharatbhai Patel, Building scalable business intelligence systems in the cloud: A technical approach, International Journal of Science and Research Archive, 2021, 02(01), 212-215., Received on 03 February 2021; revised on 17 April 2021; accepted on 19 April 2021,DOI: https://doi.org/10.30574/ijsra.2021.2.1.0047
4. Dip Bharatbhai Patel, Understanding the Impact of Text Analytics on Social Media Sentiment Analysis, Vol. 5 No. 5 (2019) ,September-October, DOI: https://doi.org/10.32628/CSEIT1936413
5. Dip Bharatbhai Patel, Real-Time Data Streaming in BI Dashboards: Opportunities and Challenges, Archives / Vol. 5 No. 4 (2019) : July-August, DOI: https://doi.org/10.32628/CSEIT192142
6. Dip Bharatbhai Patel, the Role of Data Visualization in Modern Intelligence, Archives / Vol. 2 No. 2 / Research Article, Published, 2020-11-28 DOI: https://doi.org/10.32996/jcsts.2020.2.2.4
7. A. Srivastava, "AI-assisted cloud migration of healthcare claims data from legacy systems: A metadata-driven framework," Int. J. Commun. Networks Inf. Secur., vol. 14, no. 3, Nov. 2022. [Online]. Available: https://ijcnis.org/index.php/ijcnis/article/view/8682
8. A. Srivastava, "Enhancing provider and claims data accuracy using artificial intelligence on cloud-native data platforms," J. Comput. Anal. Appl., vol. 31, no. 4, Nov. 2023. [Online]. Available: https://eudoxuspress.com/index.php/pub/article/view/4347/3189
9. A. Srivastava, "Optimizing large-scale data migration for cloud-native architectures," J. Comput. Anal. Appl., vol. 31, no. 2, pp. 652–662, May 2023. [Online]. Available: https://eudoxuspress.com/index.php/pub/article/view/4603
10. Backup Integrity and Recovery Readiness Assessment for High-Availability Databases Raghu Gollapudi, Computer Fraud & Security, Vol. 2023, Issue 8, pp. 35-48, 10.52710/cfs.1031

11. Risk-Controlled Near-Zero-Downtime Oracle Database Migration Using GoldenGate Raghu Gollapudi, International Journal of Computational and Experimental Science and Engineering, 8(3), 2022 ,10.22399/ijcesen.5382
12. Manoj Kumar Kagitha, Automated Security Compliance in Ci/Cd: A Natural Language Processing (NLP) Approach to Detecting Vulnerabilities in Infrastructure-As-Code (Terraform/Bicep),Publication year 2021, Volume 2021, Issue 12, https://computerfraudsecurity.com/index.php/journal/article/view/970
13. Manoj Kumar Kagitha, Self-Healing Cloud Infrastructure: A Hybrid Reinforcement Learning Framework for Predicting Microservice Failures in Azure Kubernetes Service (AKS),Publication year 2023, Volume 2023, Issue 12,https://computerfraudsecurity.com/index.php/journal/article/view/971
14. Manoj Kumar Kagitha, LOAD BALANCING TECHNIQUES IN DISTRIBUTED CLOUD ENVIRONMENTS, (IJESMR - 2017), D.O.I:10.53555/jcr.v6:i1.13380, https://jcreview.com/archives/volume-6/issue-1/13380
15. Manoj Kumar Kagitha, AI-DRIVEN OBSERVABILITY FOR HYPERSCALE COLOCATION DATA CENTERS: A CASE STUDY OF LOG ANALYTICS AND ANOMALY DETECTION PIPELINES, Archives / Vol. 27 No. 7 (2019): JOCAAA,https://www.eudoxuspress.com/index.php/pub/article/view/5053
16. Manoj Kumar Kagitha, GREENOPS IN THE CLOUD: AN AI-DRIVEN TELEMETRY ANALYSIS MODEL FOR REDUCING CARBON FOOTPRINT IN HIGH-AVAILABILITY CLUSTERS, Vol. 28 No. 6 (2020): JOCAAA, https://www.eudoxuspress.com/index.php/pub/article/view/5054
17. Naresh Lokiny. (2022). Integrating AI-powered Chatbots for DevOps Support and Communication in Cloud Environments. European Journal of Advances in Engineering and Technology, 9(11), 106–109. https://doi.org/10.5281/zenodo.13325989
18. Naresh Lokiny, (2021), "Disaster Recovery and Business Continuity Planning in DevOps Cloud with AI", International Journal of Science and Research (IJSR), 10(3), 2024-2027. https://dx.doi.org/10.21275/SR24724151733, https://www.ijsr.net/getabstract.php?paperid=SR24724151733
19. Naresh Lokiny, & Ranganath Nandanampati. (2020). DevSecOps: Integrating Security into DevOps with AI in Cloud. Journal of Scientific and Engineering Research, 7(10), 239–242. https://doi.org/10.5281/zenodo.13348695
20. Naresh Lokiny, & Pradip Reddy. (2021). Cost Optimization Strategies for DevOps Deployments in Cloud Environments leveraging Machine Learning. European Journal of Advances in Engineering and Technology, 8(3), 69–72. https://doi.org/10.5281/zenodo.13325845
21. Jayanth Para, AI Based Cloud Computation Observational Method & Process, Vol. 51 No. 4 (2023): October-December 2023, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/171 , DOI: https://doi.org/10.46121/pspc.51.4.3
22. Jobayar Alom, Ahsan Ahmed. (2023). Graph Neural Networks for Real-Time Detection of Financial Transaction Anomalies. Acta Scientiae, 24(5), 82–90. https://doi.org/10.22178/acta.24.5.6
23. Kaleshwar Aryasomayajula, PREVENTING BIAS IN AI-BASED DECISION-MAKING: ANALYZING TECHNIQUES TO REMOVE UNFAIR PREJUDICE IN ALGORITHMIC OUTCOMES, Vol. 50 No. 2 (2022): April-June 2022, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/294
24. Kaleshwar Aryasomayajula, MODERN DEVELOPMENT PRACTICES: INVESTIGATIONS INTO SERVERLESS COMPUTING, LOW-CODE/NO-CODE PLATFORMS, AND DEVELOPER PRODUCTIVITY IN REMOTE ENVIRONMENTS, Vol. 50 No. 4 (2022): October-December 2022, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/364
25. Vikas Suresh Bagora, KERNEL-LEVEL PERFORMANCE OPTIMIZATION FOR CARRIER-GRADE BROADBAND AND HIGH-SPEED NETWORKING SYSTEMS, Vol. 47 No. 1 (2019), Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/359
26. Vikas Suresh Bagora, SCALABLE 128G FIBRE CHANNEL AND ETHERNET CONVERGENCE FOR NEXT-GENERATION DATA CENTER NETWORKS, Vol. 50 No. 4 (2022): October-December 2022, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/362
27. Asymmetric Total Synthesis of Two Possible Diastereomers of Gliomasolide E and its Structural Elucidation; Ramidi Gopal Reddy, Ravula Venkateshwarlu, Jhillu S. Yadav and Debendra K.

Mohapatra. (J. Org. Chem. 2017, 82(2), 1053-1063). https://doi.org/10.1021/acs.joc.6b02611 , https://pubs.acs.org/doi/abs/10.1021/acs.joc.6b02611

28. Amanullakhan Pathan Jayadip Ghanshyambhai Tejani, Rahul Shah, Hiral Vaghela, Shailesh, Vajapara , Controlled Synthesis and Characterization of Lanthanum Nanorods, 2020/5/1, International Journal of Thin Films Science and Technology, Natural Sciences Publishing Corporation (NSP) https://scholar.google.com/citations?view_op=view_citation&hl=en&user=efbNMkwAAAAJ & citation_for_view=efbNMkwAAAAJ:M3NEmzRMIkIC
29. Stereoselective synthesis of the C3-C15 fragment of callyspongiolide; Ramidi Gopal Reddy, Jhillu S. Yadav and Debendra K. Mohapatra. (Tetrahedron Letters. 2018, 59, 3579-3582). https://doi.org/10.1016/j.tetlet.2018.08.041, https://www.sciencedirect.com/science/article/pii/S0040403918310426
30. Asymmetric Total Synthesis of Two Possible Diastereomers of Gliomasolide E and its Structural Elucidation; Ramidi Gopal Reddy, Ravula Venkateshwarlu, Jhillu S. Yadav and Debendra K. Mohapatra. (J. Org. Chem. 2017, 82(2), 1053-1063). https://doi.org/10.1021/acs.joc.6b02611, , https://pubs.acs.org/doi/abs/10.1021/acs.joc.6b02611
31. Kaleshwar Aryasomayajula, PREVENTING BIAS IN AI-BASED DECISION-MAKING: ANALYZING TECHNIQUES TO REMOVE UNFAIR PREJUDICE IN ALGORITHMIC OUTCOMES, Vol. 50 No. 2 (2022): April-June 2022, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/294
32. Kaleshwar Aryasomayajula, MODERN DEVELOPMENT PRACTICES: INVESTIGATIONS INTO SERVERLESS COMPUTING, LOW-CODE/NO-CODE PLATFORMS, AND DEVELOPER PRODUCTIVITY IN REMOTE ENVIRONMENTS, Vol. 50 No. 4 (2022): October-December 2022, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/364
33. Kaleshwar Aryasomayajula, Micro services: “A Comparative Analysis of Monolithic vs. Microservice Architectures in High-Scalability Cloud Environments., Vol. 51 No. 2 (2023): April-June 2023 , Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/374
34. Controlled Synthesis and Characterization of Lanthanum Nanorods, Author(s), Jayadeep Tejani, Rahul Shah, Hiral Vaghela, Shailesh Vajapara, Amanullakhan Pathan, doi:10.18576/ijtfst/090205, URL: https://www.naturalspublishing.com/Article.asp?ArtcID=20705, May-2020
35. Sudipkumar Ghanvat , Aishwarya Badlani, Shreya Sandeep Joshi, Marketing Effectiveness in the Post-Cookie Era: A Comparative Analysis of First Party and Zero-Party Data Strategies on Campaign Performance and Customer Equity, Vol. 31 No. 4 (2023): JOCAAA , https://www.eudoxuspress.com/index.php/pub/article/view/3940
36. Chander Vijay S Sanbhi, BAYESIAN UPDATING OF RAM MODELS FOR DYNAMIC AVAILABILITY FORECASTING UNDER REALISTIC DOWNTIME CONSTRAINTS, Vol. 51 No. 3 (2023): July-September 2023, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/381
37. Chander Vijay S Sanbhi, HYBRID RELIABILITY MODELLING FOR ROTATING EQUIPMENT: PHYSICS- INFORMED LEARNING WITH SPARSE FAILURE DATA, Vol. 51 No. 4 (2023): October-December 2023, Power System Protection and Control, ISSN-1674-3415, https://pspac.info/index.php/dlbh/article/view/382
38. A. Srivastava, "AI-assisted cloud migration of healthcare claims data from legacy systems: A metadata-driven framework," Int. J. Commun. Networks Inf. Secur., vol. 14, no. 3, Nov. 2022. [Online]. Available: https://ijcnis.org/index.php/ijcnis/article/view/8682
39. A. Srivastava, "Enhancing provider and claims data accuracy using artificial intelligence on cloud-native data platforms," J. Comput. Anal. Appl., vol. 31, no. 4, Nov. 2023. [Online]. Available: https://eudoxuspress.com/index.php/pub/article/view/4347/3189
40. A. Srivastava, "Optimizing large-scale data migration for cloud-native architectures," J. Comput. Anal. Appl., vol. 31, no. 2, pp. 652–662, May 2023. [Online]. Available: https://eudoxuspress.com/index.php/pub/article/view/4603

41. Santhosh Chitraju Gopal Varma, The Evolution of Cloud-Native Architectures: Exploring the Synergy between Kubernetes and Microservices, DOI: https://doi.org/10.63282/3050-9246.IJETCSIT-V1I4P104, Vol. 1 No. 4 (2020),

https://ijetcsit.org/index.php/ijetcsit/article/view/108
42. Santhosh Chitraju Gopal Varma, Artificial Intelligence in Cloud Computing: Building Intelligent, Distributed, and Fault-Tolerant Systems, Vol. 3 No. 1 (2022), DOI: https://doi.org/10.63282/3050-9416.IJAIBDCMS-V3I1P105, https://ijaibdcms.org/index.php/ijaibdcms/article/view/78
43. Sunil Karmakar, AI driven Identity & Access Management with Oracle Cloud security, Vol. 30 No. 2 (2022): JOCAAA, https://eudoxuspress.com/index.php/pub/article/view/4118
44. Bandari, V. V., & Lekkala, H. B. (2024). Physics-informed reinforcement learning for real-time control of complex manufacturing processes. Power System Protection and Control, 52(2), 164–170. https://pspac.info/index.php/dlbh/article/view/343
45. Bandari, V. V., & Lekkala, H. B. (2024). AI-based operator behavior monitoring and cost optimization using digital traceability in manufacturing systems. Power System Protection and Control, 52(1), 38–45. https://pspac.info/index.php/dlbh/article/view/342
46. Kancharla Vijayalakshmi, Bethapudi Samuel Anand Andrews, Estimation of Cyproheptadine Hydrochloride and Tricholine Citrate Simultaneously in Syrup Dose by Applying Stability Indicating HPLC Methodology, December 31, 2020, Vol. 11, Issue 4, Pages 1813–1819, DOI: 10.26452/ijrps.v11iSPL4.4383, https://ijrps.com/v11si4/3683
47. Vijayalakshmi Kancharla ,Kumar Abbaraju V D N, Andrews B S A, Nagaraju Dasari, Nagu Korupolu, Melphan as Anticancer and New Insights of Drug as Parenteral Dosage form against SARS-CoV-2 by Using Development and Validation , November 18, 2020, Vol. 11, Issue 1, Pages 1437–1446
DOI: 10.26452/ijrps.v11iSPL1.3686, https://ijrps.com/v11si1/2911
48. Ramesh, D., Singh, A., & Channaiah, L. H. (2024). The importance of validated kill steps and thermal death kinetics of Salmonella in food safety management. Global Journal of Nutrition & Food Science, 5(1), Article 000601. https://doi.org/10.33552/GJNFS.2024.05.000601
49. Bharati Reddi, Chandan Kishor,Aruna Jangam, Sandeepchowdary Bala, Uma Rajeswari Batchu ,Rambabu Gundla , Anthony Addlagatta, Regioselectivity in inhibition of peptide deformylase from Haemophilus influenzae by 4- vs 5-azaindole hydroxamic acid derivatives: Biochemical, structural and antimicrobial studies, Date: November 2022, Article: 106095, Volume: Volume 128, 10.1016/j.bioorg.2022.106095, https://www.sciencedirect.com/science/article/abs/pii/S0045206822005016
50. Bhavita Kattula, Bharati Reddi, Aruna Jangam, Lekhika Naik , Bala Manikanta Adimoolam , Suresh Vavilapalli , Sayanna Are , Jagadeshwar Reddy Thota, Surender Singh Jadav , Mohammed Arifuddin Anthony Addlagatta, Development of 2-chloroquinoline based heterocyclic frameworks as dual inhibitors of SARS-CoV-2 MPro and PLPro, Date: 1 July 2023 Article: 124772, Volume: Volume 242, Part 2, 10.1016/j.ijbiomac.2023.124772 https://www.sciencedirect.com/science/article/pii/S0141813023016665
51. Iram Irfan, Asghar Ali, Bharati Reddi, Mohd. Abrar Khan, Phool Hasan, Sarfraz Ahmed Amad Uddin, Magdalena Piatek, Kevin Kavanagh, Mohammad Abid, Design, Synthesis and Mechanistic Studies of Novel Isatin-Pyrazole Hydrazone Conjugates as Selective and Potent Bacterial MetAP Inhibitors, 19 August 2022, https://doi.org/10.3390/antibiotics11081126 https://www.mdpi.com/2079-6382/11/8/1126
52. GAURAV KUMAR SINHA, Efficient Test-Time Compute Scaling for Resource-Constrained Large Language Models Vol. 32 No. 1 (2024): JOCAAA, DOI: https://doi.org/10.1109/ICAC3N56670.2022.10074458 https://eudoxuspress.com/index.php/pub/article/view/5918
53. GAURAV KUMAR SINHA , The Impact of Artificial Intelligence on Predictive Quality Assurance in Modern Manufacturing, Vol. 33 No. 05 (2024): JOCAAA, https://doi.org/10.1109/ICAC3N56670.2022.10074458 , https://eudoxuspress.com/index.php/pub/article/view/5919
54. GAURAV KUMAR SINHA , Hybrid Quantum-Classical Neural Networks for Combinatorial Optimization and Machine Learning, Vol. 33 No. 08 (2024): JOCAAA, DOI: https://doi.org/10.1109/ICAC3N56670.2022.10074458, https://eudoxuspress.com/index.php/pub/article/view/5921
55. Lalith Chandra Bandaru , Mohammed Shakeer Bandrevu, GRAPHSEC: Graph-Based Supply Chain Attack Detection and Risk Propagation Analysis for Enterprise Salesforce Deployment Pipelines, Paper Id: 233135,Published On: 2023-11-03, Published In: Volume 11, Issue 6, November-December 2023, https://www.ijirmps.org/research-paper.php?id=233135 , DOI: https://doi.org/10.37082/IJIRMPS.v11.i6.233135