processes

MDPI

Article

A Large Language Model-Based Manufacturing Execution System for Heat Treatment and Its Evaluation Method

Yixiao Sun [1], Wenyu Zhang [1,*], Chao Liu [2], Xiaohu Deng [3] and Dongying Ju [2,4,*]

[1] School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan 114051, China; 202081203562@stu.ustl.edu.cn
[2] Zhejiang XCC Group Co., Ltd., Xinchang 312500, China; emacs157@yahoo.com
[3] School of Mechanical Engineering, Tianjin University of Technology and Education, Tianjin 300222, China; dengxh@tute.edu.cn
[4] College of Mechanical and Electrical Engineering, Beijing Information Science & Technology University, No. 55 Taihang Road, Changping District, Beijing 102206, China
* Correspondence: zhangwenyu8518@ustl.edu.cn (W.Z.); dyju.sitec@gmail.com (D.J.)

Abstract

Quality investigations in heat treatment require inspection results to be linked with processing histories, while conventional manufacturing execution systems distribute these records across separate modules. This study develops a read-only language-model interface with product and basket-time linkage and evaluates query execution separately from evidence correctness. A same-model comparison using GPT–OSS 20B covers 48 tasks and 288 attempted executions, with 273 complete responses. Frozen condition-screen counts were similar; asymmetrically reviewed scope corrections are not used as comparative evidence. Independent backend checks identified alarm-count inconsistencies before evidence reached the model. A retrospective, non-blinded review of 41 numerical fallbacks, assisted by artificial intelligence (AI), found six pre-fallback answers supported relative to the saved evidence, 34 containing unsupported or incorrect claims, and one indeterminate. These assessments do not independently validate the enterprise records. Three quality cases illustrate linkage to stage-specific furnace evidence, but incomplete basket coverage leaves the relationship between temperature deviations and hardness nonconformity undetermined. The study separates returned states, answer support, and independently verified facts when evaluating language-model access to manufacturing records.

Keywords: manufacturing execution system; large language model; heat treatment; tool calling; quality traceability

Check for updates

Academic Editor: Shen-Haw Ju

Received: 21 September 2026
Revised: 24 October 2026
Accepted: 27 October 2026
Published: 6 November 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Manufacturing execution systems (MES) connect enterprise planning with shop-floor production and provide records for quality management and process traceability [1]. In heat treatment, investigating a failed hardness inspection requires more than retrieving the inspection result. Engineers must identify the processed product, recover its loading and processing intervals, and compare measured furnace conditions with their targets. These records are commonly distributed across inspection, product-tracking, equipment, and process-history modules. Their practical value therefore depends on preserving the identity and time relationships that connect a quality measurement to its processing context.

Natural-language access offers a way to retrieve these records without requiring users to know every interface or database field. Large language models (LLMs) have been applied to

manufacturing knowledge retrieval, planning, maintenance, and quality management [2,3]. Retrieval-augmented generation supplies external information for response generation [4–6], while reasoning-and-action methods and tool learning support the invocation of external functions [7,8]. Such capabilities make a language model a potential interface to manufacturing services, but successful invocation alone does not establish that the requested entity and time interval were preserved or that the returned evidence is complete.

The closest manufacturing-execution studies address different parts of this problem. Choi and Jeong [9] describe a domain-specific architecture combining retrieval and text-to-SQL (Structured Query Language) generation. Yuan et al.'s Chat with MES [10] uses request rewriting and multistep operation generation, evaluated with manually designed requests in a simulated garment MES. Fan et al. [11] combine a language-model interface with reinforcement-learning support for photoresist viscosity adjustment. These studies establish substantial prior work in conversational access and orchestration; generic tool calling is not a contribution unique to the present study. Here, the focus is the connection between inspection and thermal-history records in an operating heat-treatment MES, as well as the distinction between query-condition coverage, faithful explanation of tool output, and agreement with the underlying enterprise records. Table 1 summarizes this positioning without comparing scores obtained on different tasks.

Table 1. Positioning relative to closely related manufacturing execution system (MES) studies. SQL denotes Structured Query Language.

Study	Main Mechanism and Setting	Relation to the Present Investigation
Choi and Jeong [9]	Domain-specific retrieval and text-to-SQL architecture	Provides architectural context for natural-language access to structured manufacturing data.
Yuan et al. [10]	Request rewriting and multistep operations in a simulated garment MES	Establishes the tool-orchestration baseline; the present work examines record linkage and evidence verification in heat treatment.
Fan et al. [11]	Language-model orchestration of computational and reinforcement-learning services for viscosity adjustment	Addresses parameter recommendation and control support rather than retrospective inspection-to-thermal-history linkage.
Present study	Read-only MES queries, basket-time linkage, and a same-model component comparison	Evaluates query execution and evidence provenance through paired task conditions and linked inspection histories.

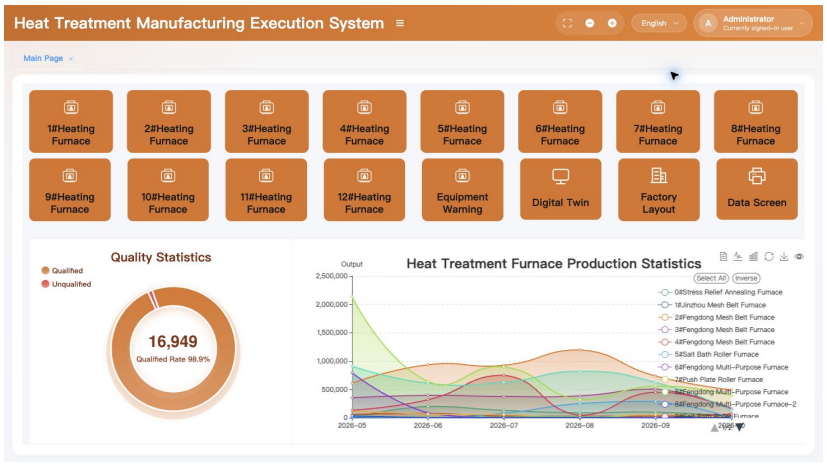
Other manufacturing applications include natural-language production planning, carbon-emission management, multi-agent coordination, and digital-twin parametrization [12–17]. Human-in-the-loop approaches retain engineering oversight [18]. In quality management and maintenance, cognitive assistants, causal knowledge, asset information models, and troubleshooting structures organize domain evidence [19–24]. Time-series and multimodal approaches support equipment fault diagnosis [25–30]; visual- and vision-language approaches address industrial inspection [31–36]. These methods motivate evidence-based assistance but do not replace inspection specifications or the recorded links between products and process histories. Hallucination research further motivates checking claims against their sources [37].

This study examines how a language-model interface can support heat-treatment quality investigation through existing read-only services. The main contributions are as follows. First, the implemented calling and response-processing rules are specified so that

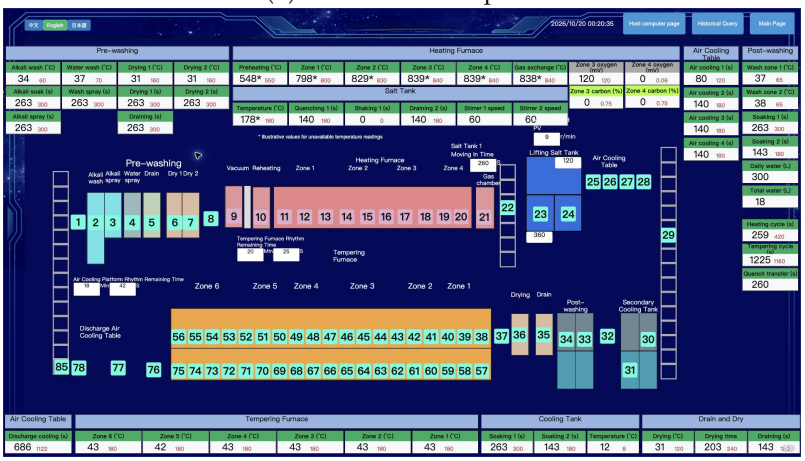
schema checks, result-state handling, and numerical screening can be evaluated separately from business-semantic correctness. Second, inspection, product, basket, and furnace records are linked through explicit identifiers and timestamps, with three quality cases used to examine the interpretation of the retrieved thermal evidence. Third, a same-model comparison and task-cluster analysis distinguish query execution from evidence correctness and identify failures that would be hidden by aggregate workflow scores. No claim is made that generic orchestration itself is novel or that these experiments demonstrate improved user efficiency.

2. Methods

The method connects natural-language requests to read-only MES services and assembles the returned records for engineering review. Its two components are a bounded tool-calling procedure and an inspection-to-process evidence chain. The evaluation follows the same separation: it checks query conditions and response completion, then examines evidence provenance through independent backend comparisons and quality-case records. Figure 1 shows the application environment in which the relevant records are maintained.



(**a**) MES main workspace



(**b**) Shop-floor process monitoring

Figure 1. Representative interfaces of the implemented heat treatment manufacturing execution system (MES). (**a**) Furnace access, quality statistics, and production trends; colors distinguish furnace series in the production chart. (**b**) Process-stage status, workpiece positions, and measured and target parameters. Numbered markers indicate displayed workpiece positions; large black values show current readings and smaller red values show targets. Asterisked temperatures are illustrative display values for unavailable readings, not experimental measurements.

https://doi.org/10.3390/pr14213597

2.1. Read-Only Tool Calling and Response Processing

For request q, the model selects tools and proposes arguments from a complete catalog. The response is formulated using the returned evidence, as expressed in Equation (1):

$$y = f_\theta(q, E), \qquad E = \bigcup_{k=1}^{K} g_k(x_k). \tag{1}$$

Here, $g_k(x_k)$ denotes the records returned by the kth call, and f_θ denotes language generation. The calling layer parses JavaScript Object Notation (JSON) arguments and invokes permitted read-only functions. It does not give the model unrestricted database or equipment-control access. Figure 2 shows the overall architecture. Table 2 specifies the evaluated schema checks, result-state handling, and numerical screening; general-purpose identifier and temporal-semantic validation remain unverified.

The model is served through an internal inference endpoint, separately from the client. Source inspection identifies collectors for furnace measurements and basket timestamps, MySQL-based collector inputs, and a .NET MES data-access backend. In the application path described by the source code, collectors read historical shop-floor records, which are stored in the MES and exposed through its application programming interfaces (APIs). The Model Context Protocol (MCP) service retrieves these records and supplies them to the model context. Saved service responses verify particular query boundaries, not the complete deployed acquisition path. The study did not instrument the sensor-to-controller link or verify the remote deployment revision, so no specific fieldbus or sensor protocol is assumed.

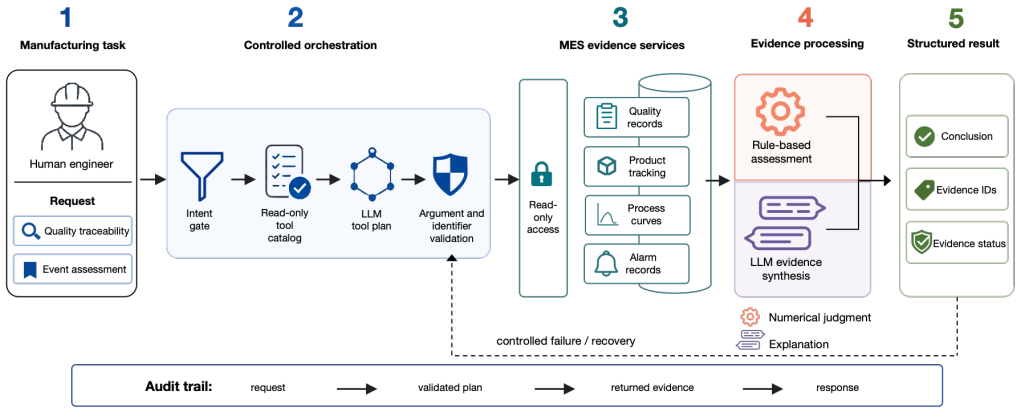


Figure 2. Overall architecture of the read-only manufacturing execution system (MES) interface. The large language model (LLM) selects tools, MES services retrieve manufacturing records, and rule-based numerical summaries support evidence explanation. Numbers 1–5 identify the manufacturing task, controlled orchestration, evidence services, evidence processing, and structured result, respectively. Solid arrows show the main flow; the dashed arrow denotes controlled failure and recovery. ID denotes identifier.

Algorithm 1 defines the evaluated component configuration. Three result states are distinguished by the adapter and evaluator: evidence, no matching records, and tool error. A safe explanation of an error is not credited as successful retrieval. Tool text, sometimes containing its own structured status, is appended to the conversation; a separate uniform status field is not added to every model message. After a final answer, wholly unusable tool results trigger the error fallback before numerical screening is considered. Both original and fallback answers are retained. The ordinary configuration uses the same model, catalog, read-only interfaces, and execution budgets, without the evaluated schema, adaptation, and response-fallback package.

Table 2. Implemented checks and their interpretation in the component comparison. JSON denotes JavaScript Object Notation.

Stage	Implemented Operation	What Is Not Established by This Check
Before invocation	Confirm catalog membership, parse JSON, and check required fields and declared types	A schema-valid date or identifier can still change the requested scope.
After retrieval	Adapt the returned representation and classify evidence, no matching records, or tool error	An empty representation is not independent proof that the backend has no records.
Before presentation	Screen numerical tokens against the request and tool text; use a tool-record fallback when triggered	Token occurrence does not establish field identity, units, causal support, or factual accuracy.
Audit	Retain the proposed arguments, original and adapted returns, pre-fallback and final answers, and timings	Log completeness does not independently validate the underlying business facts.

Algorithm 1 Bounded tool calling with explicit result states.

Input: request q, catalog C, schema check V_s, budgets N, B
Output: final response y, original records E, execution log A

Initialize conversation D with the common system prompt and q;
for $r = 1, \ldots, N$ within call budget B **do**
  Request a model response using D and complete catalog C;
  if no tool call is proposed, retain the raw answer and **break**;
  for each proposed call (g, x) **do**
    Parse x; on parsing failure, retain the exception and end incomplete;
    If $g \notin C$ or V_s fails, return error text; otherwise call and adapt;
    append returned text to D; retain raw/adapted output and timings;
Classify returns as evidence, no records, or error for assessment;
If all returns are unusable, use error fallback; else apply numeric screen;
If no final answer is obtained, retain the failure reason and all timings;
Return y, E, A, keeping error explanation distinct from retrieval success.

2.2. Inspection-to-Process Evidence Linkage

The starting point is an inspection record containing an item, measured values, an acceptance range, and a product-tracking association. The tracking card supplies product, work-order, heat-number, and furnace context. A heat-treatment basket is a container holding parts during furnace processing; its process record provides timestamps for the relevant stages. As illustrated in Figure 3, the inspection record is linked to basket process records through the product-tracking card. The basket timestamps then delimit the furnace samples retrieved for analysis. The evidence set is defined in Equation (2):

$$E_Q = \{e_{\text{inspection}}, e_{\text{product}}, e_{\text{basket}}, e_{\text{curve}}, e_{\text{alarm}}\}. \quad (2)$$

A missing component is recorded rather than replaced by a model-generated value. Inspection measurements are linked to a processing context, not assigned an individual temperature history unless a specimen-to-basket correspondence exists.

Variable retrieval and numerical calculations are performed by the services and analysis scripts. The original curve service enumerates available process zones and returns measured and target values; the reported heating zones were selected for examination from these returned fields. The model did not autonomously discover the two zones by searching an unrestricted variable space. Its role is to select documented retrieval functions and explain the returned evidence. Deterministic product and timestamp linkage remains outside the model.

For valid paired measured and target samples $(x_{z,j}, s_{z,j})$ in zone z, the descriptive mean deviation and fraction below a chosen analysis threshold δ are given by Equation (3):

$$\bar{d}_z = \frac{1}{n_z}\sum_{j=1}^{n_z}(x_{z,j} - s_{z,j}), \qquad P_z(\delta) = \frac{100}{n_z}\sum_{j=1}^{n_z}\mathbf{1}\{x_{z,j} < s_{z,j} - \delta\}. \tag{3}$$

The denominator counts retrieved valid samples, not independent experiments. The original case uses a linked production interval reconstructed from eight basket records. The supplementary basket-specific audit instead uses zone-1 timestamps `keytime9--keytime10` and zone-2 timestamps `keytime10--keytime11`, followed by deduplication by zone and timestamp. These two interval definitions are reported separately. The 5 °C threshold is an analytical comparison, not a verified plant alarm limit; its sensitivity is examined in the results.

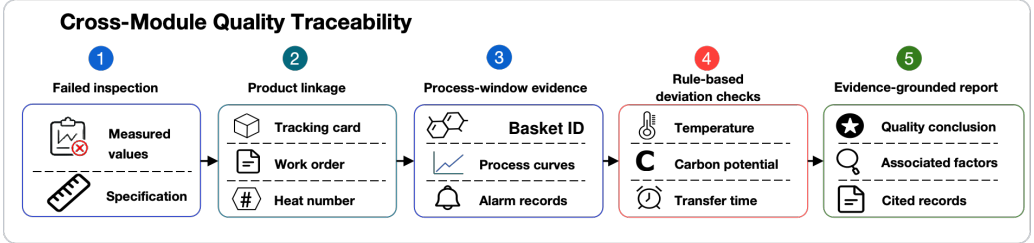


Figure 3. Quality-traceability workflow. Inspection and product associations identify basket records and process evidence for specification comparison and an evidence-based report. Numbers 1–5 denote failed inspection, product linkage, process-window evidence, rule-based deviation checks, and the evidence-grounded report. Arrows indicate the progression of evidence retrieval and assessment. ID denotes identifier.

2.3. Task-Level Evaluation

Evaluation separates the selection of retrieval functions, coverage of reference query conditions, completion of a final response, and factual agreement with independent records. Reference tools and arguments were constructed by the study authors from task requests and documented interface semantics, not from a database-derived reference answer for every field. The frozen screen permits defined alias and month-format normalization. The separate post hoc scope review permits consecutive monthly calls whose union equals the requested interval and uses the inspected energy-routing rule, where nonzero day takes precedence. Two component proposals were additionally corrected using returned dates without an equivalent blinded review of the ordinary arm. This asymmetric review is used for diagnostic interpretation only, not to estimate between-arm differences or intervals. Neither screen establishes complete conditional-chain or business-answer correctness.

Let $z_{i,r,m}$ be a binary execution measure for task i, repetition r, and configuration m. Comparisons first average the three repetitions within a task and then compare the paired task means, as defined in Equation (4):

$$\widehat{\Delta} = \frac{1}{I}\sum_{i=1}^{I}\left(\frac{1}{3}\sum_{r=1}^{3} z_{i,r,\text{component}} - \frac{1}{3}\sum_{r=1}^{3} z_{i,r,\text{ordinary}}\right). \tag{4}$$

Descriptive 95% intervals use 10,000 task-cluster bootstrap resamples [38], preserving repeated executions and pairing. Task selection is purposive, so the intervals describe variability within this task set rather than a randomly sampled enterprise workload. Total elapsed time is retained for all attempts, including incomplete ones; model-call and tool-call durations are also recorded. Human operation and browser rendering are outside this timing measure. A separate backend comparison is required before tool-output fidelity can be interpreted as enterprise-fact correctness.

The retrospective audit reuses saved requests, tool returns, and pre-fallback answers without new model executions or backend queries. It counts explicit no-record returns separately from zero alarm counts. All 41 numerical fallbacks undergo a non-blinded record review assisted by artificial intelligence (AI): an answer is supported relative to the saved evidence only if its factual statements preserve the returned fields or justified calculations. Explicitly conditional suggestions are not treated as observed facts. An unsupported factual claim excludes the whole answer from that category; ambiguous wording is indeterminate. This is not an independent domain-expert assessment or verification of source-data truth. Per-execution states, decisions, and reasons are released in Supplementary File S1.

3. Experiments and Results

3.1. Tasks and Execution Conditions

The expanded comparison comprises the original 32 tasks and 16 additional requests, with both configurations repeated three times, giving 288 attempted task executions. The original tasks cover production records, monthly yield, alarms, nonconforming products, energy consumption, equipment status, product-tracking cards, inspections, curve-query errors, and relative dates. Additional requests cover explicit day/month/year scope, multiple periods, cross-module retrieval, missing records, unsupported inferences, and conditional follow-up. They were constructed from existing interface semantics and reuse some business parameters; they are not an independently sampled unseen industrial workload. All attempts remain in the results, including 15 without a complete final answer. Table 3 gives the task structure, with sanitized definitions and reference conditions in Supplementary File S1.

Table 3. Task composition and example requirements. Result absence and tool errors are recorded separately from task complexity.

Task Group	Number	Representative Requirement
Original functional set	32	Query a named production line and date; retrieve a product card or inspection; request monthly yield or energy; handle empty or failed retrieval.
Temporal-scope extensions	3	Preserve an explicit daily, monthly, or annual energy scope.
Multiple-query requests	2	Compare two periods without silently dropping one.
Cross-module requests	3	Combine production/alarm, card/curve, or energy/yield information.
Quality and missing-evidence probes	7	Compare inspections, preserve missing-record status, avoid unsupported conclusions, and handle conditional follow-up or an erroneous curve query.
Relative-date extension	1	Resolve yesterday using the current-time service.

Both configurations use `gpt-oss:20b`, the same system prompt, and the same 18-tool catalog. The component configuration adds the existing schema check, output adapter, and numerical/no-evidence fallback. This is a package comparison, not an ablation of each component or a test of a complete semantic validator. Runtime metadata identifies Ollama-compatible serving version 0.33.1, a 20.9-billion-parameter model using four-bit microscaling floating-point (MXFP4) weight quantization, and a loaded context of 32,768 tokens. Generation temperature is 0; output limits are 1024 tokens for the initial call and 4096 for subsequent calls, with at most five model rounds and eight tool calls. Model and tool timeouts are 90 and 45 s. Retries of model requests over the Hypertext Transfer Protocol (HTTP) are disabled; the inherited MCP client retains a closed-connection reconnect-and-reissue

branch, whose activation was not established in this run. Inference ran on an internal server equipped with an RTX 5090 graphics processing unit (GPU) and 512 GB of random-access memory (RAM). The remote MCP deployment revision was not verified. Task order is randomized with a fixed seed. The model receives the natural-language request, prompt, and tool descriptions, but no task category, reference arguments, or expected outcome.

One brief model-connectivity call preceded the schedule, without forcing a service restart or model unload; these results are not a cold-start benchmark. Archived 32-task results are reanalyzed separately from the expanded comparison. Quality evidence comes from read-only enterprise inspection, product, basket, and curve records. An initial feasibility screen of 100 recent card groups returned 451 inspection records. Separately, the latest 20 nonconforming groups supplied nine hardness candidates; continued retrieval yielded three cases with linked curves. Historical conforming references were then sought, retaining incomplete candidates in the audit. The three cases include the original case and two additions. Sources, selection bounds, and missing calibration information are documented in Supplementary File S1.

3.2. Same-Model Component Comparison

Table 4 reports the frozen screen for both configurations. Required retrieval tools were called in 144/144 ordinary and 143/144 component executions; frozen condition-screen counts were 139/144 and 138/144. The separate post hoc scope review produced 139/144 and 140/144 after two component-only corrections. The review was not blinded and did not re-examine the ordinary arm equivalently. Accordingly, the post hoc review is not used to estimate between-arm coverage differences or intervals. The frozen flags and asymmetric corrections remain separately available in Supplementary File S1; neither is an independent answer-accuracy measure.

Table 4. Expanded comparison using the frozen condition screen. The asymmetric post hoc scope corrections are excluded from this table and from comparative coverage inference. Frozen screen flags do not establish answer accuracy. Time includes all attempts.

Tasks	Configu-ration	Attempts	Frozen Screen	Complete Answer	Median Time (s)
All 48	Ordinary	144	139/144	139/144	7.6
All 48	Component	144	138/144	134/144	6.7
Original 32	Ordinary	96	96/96	94/96	6.1
Original 32	Component	96	96/96	91/96	5.9
Added 16	Ordinary	48	43/48	45/48	12.7
Added 16	Component	48	42/48	43/48	11.5

Response completion exposes a different limitation. Ordinary calling produced 139 complete answers, compared with 134 for the component configuration. Incomplete executions comprised one versus five output-length truncations and four versus five exhausted round budgets. The component package triggered 47 fallbacks: 41 numerical and six unusable-evidence fallbacks. Review of all 41 pre-fallback answers classified six as supported relative to the saved request/tool evidence (14.6%), 34 as containing an unsupported or incorrect factual claim, and one as indeterminate because a growth ratio was worded ambiguously. The six supported answers comprise two approximate production-change summaries and four query-attributed missing-record explanations; list markers or justified arithmetic triggered replacement. Other answers included invented inspection rows, wrong field/date assignments, and unsupported operational conclusions. Some erroneous answers also contained justified calculations, so trigger occurrence is not equivalent to correct error

detection. These non-blinded AI-assisted judgments neither establish independent business truth nor show that 34 errors were corrected by the fallback. The per-answer reasons and audit limits are in Supplementary File S1.

Timing also depends on the task. Added requests took approximately twice as long at the median as the original requests. For example, ordinary production requests had a median of 9.6 s, compared with 4.4 s for yield; multiple-query requests took 22.1 s. The corresponding component medians were 10.9, 4.0, and 32.8 s. Average accumulated model-call time was 9.71 and 9.09 s, compared with 0.73 and 0.71 s for tool execution. The task-paired mean total-time difference was −0.64 s, with a 95% interval from −1.77 to 0.41 s. Thus, neither a stable speed improvement nor a reduction in human task time follows from the observed medians.

3.3. Query Semantics and Independent Evidence Checks

The archived 96 executions illustrate why scoring stages must be separated. Exact dictionary matching accepted 24/96 argument sets, whereas alias and scope normalization accepted 95/96; the task-cluster interval for the latter is 96.9–100%. The remaining failure added `day=1` to a monthly energy request. The frozen no-record flag identified 18 executions; inspection of saved text identifies six additional explicit production-card absence returns, giving 24/96 returned absences. These six wording cases also explain the original 89/96 automatic versus 95/96 post hoc workflow-screen acceptance. A statement faithful to an absence return is not proof that the backend lacked records. No matched contemporaneous source snapshots establish the business truth of these 24 historical absences, and the archived cohort is separate from the expanded 288 executions. Scorer independence and agreement cannot be reconstructed; no independent human panel is claimed. The archived response-time median remains 6.7 s, with a task-cluster interval of 5.52–7.73 s.

Constructed regression probes confirm the limits of the existing checks. Schema validation accepts a monthly action with a nonzero day and does not itself reject an empty identifier. An auxiliary normalizer repairs the known historical month/day proposal in offline replay; it is not part of the frozen experimental configuration and was not evaluated in a live comparison. In a separate replay of 50 saved energy proposals, one argument set changed with no scope correction or regression. The rule reads the proposed action rather than the user's full request. Numerical occurrence checks can accept an unsupported value 3 when only 30 occurs, and can miss an incorrect unit or unsupported nonnumerical inference. Conversely, a list number can trigger a false alarm. These outcomes explain why structured fields and source attribution need to be checked before interpreting a workflow score.

Independent backend checks revealed a defect in the system under study. For six requests, direct backend queries bracketed an identical MCP alarm call. Four stable backend counts were 51, 62, 113, and 16 while MCP returned zero; two zero-count requests agreed. A second read-only inspection reproduced the discrepancy, but the deployed conversion revision and the defect's scope across interfaces were not established. These are counts of records in the backend occurring state, not incident counts. The checks are separate from the 288 executions and do not retrospectively label the historical model answers.

Across the expanded 288 executions, 83 contain at least one no-record return: 42 ordinary and 41 component. The last tool return is no-record in 39 executions per arm (78 total); this is a tool-sequence endpoint, not a verified final-answer state. There are 55 and 60 such tool returns, respectively. A further 42 alarm calls (21 per arm) return zero counts but are classified as evidence rather than explicit no-records. Thus, an explicit-state count alone does not bound the defect. The saved logs permit recounting of these representations but provide no matched contemporaneous backend snapshot establishing absence for each affected execution. The six separate alarm checks do not validate other

interfaces or all historical zero returns. Absence correctness and the prevalence of false absence therefore remain undetermined; returned-state recognition is not reported as verified business-result accuracy.

A conditional card-to-inspection request further exposed an identifier-interface ambiguity. The tool returned a product barcode but no separately labelled internal tracking-card identifier (ID). Ordinary calling issued a subsequent inspection query in all three repeats, using the barcode and other returned business fields; the component configuration stopped after the card query in all three repeats. Because the request conditioned follow-up on an ID and the tool description also described its ID field as accepting a barcode, neither the number of calls nor the presence of an answer is a sufficient reference for the whole chain. The trace is released as a documented example of intermediate-evidence-dependent behavior, not as a successful autonomous investigation. Explicitly distinguishing internal IDs from business barcodes is required before scoring this task as a conditional retrieval success.

3.4. Quality Cases and Historical Conforming References

Case `REAL-Q-2C7883C8` links nine failed surface-hardness values on the Vickers hardness (HV) scale to eight basket process records and a 419.3-min production interval containing 432 furnace records. All hardness values were below the 715–780 HV range: 683–707 HV, with a mean of 696.7 HV. Over the linked interval, zone 1 averaged 796.7 °C against 820 °C, and zone 2 averaged 824.1 °C against 835 °C. Fractions more than 5 °C below target were 69.9% and 61.1%, respectively (Figure 4). These sustained deviations identify records worth examining when investigating the failed inspection. Available historical evidence does not independently establish the alarm state for this processing interval.

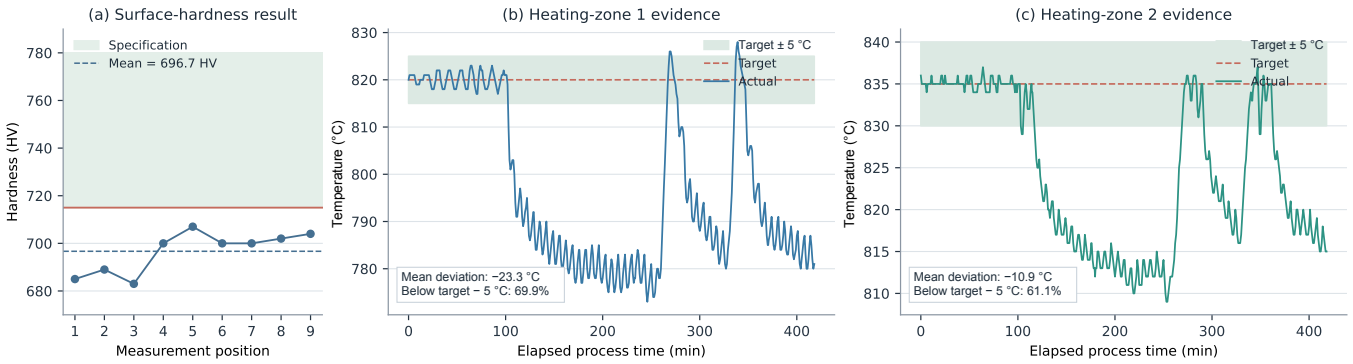


Figure 4. Original anonymized quality case. (**a**) Nine Vickers hardness (HV) values and specification limits. (**b**,**c**) Measured and target temperatures over the linked production interval. These are interval-associated process samples, not nine specimen-specific thermal histories.

Table 5 adds two cases and historical conforming references matched on product identity, specification, inspection item, and a verified common physical furnace within each pair. The HV addition contains three of nine values below 715 HV, and the Rockwell C-scale hardness (HRC) addition contains two below 60 HRC, although both means lie within their respective limits. Thus, a comparison of average hardness alone would overlook the recorded local nonconformities. HV and HRC are analyzed separately.

Basket-specific retrieval confirmed that nonempty heating samples lay within the requested basket's stage timestamps. For C3, the union of the eight stage windows contained 61 unique zone-1 and 69 zone-2 samples, with mean deviations of −23.9 and −13.1 °C. These stage-specific subsets differ in temporal scope from the 432-point production interval shown in Figure 4. Requests were bounded to 12 basket records per card, and a 100-record list may be truncated; Table 5 therefore reports observed coverage rather than complete batch populations. Individual inspection positions could not be assigned to specific baskets.

Table 5. Three quality cases and historical references. C3 corresponds to the case in Figure 4. Each fraction is checked/returned basket records; the two fractions distinguish case and reference, respectively. A capped list may omit records. References are observational, not controlled trials. HV denotes Vickers hardness; HRC denotes Rockwell hardness on the C scale.

Case	Specification	Failed Inspection	Below Lower Limit	Reference Values	Basket Coverage: Case; Reference
C1	715–780 HV	704–736; mean 720.8	3/9	732–741; mean 735.9	12/≥100; 12/≥100
C2	60–64 HRC	59.8–60.8; mean 60.3	2/9	61.0–61.7; mean 61.4	8/8; 12/21
C3	715–780 HV	683–707; mean 696.7	9/9	730–741; mean 735.2	8/8; 12/74

Within the checked subsets, C2 has zone deviations of −20.7 and −16.2 °C, compared with −1.9 and −0.7 °C in its reference, with different target settings. C3's checked conforming-reference samples have a zone-1 mean deviation of −39.2 °C. These are observations within retrieved stage windows. Given the capped and unequal basket coverage in Table 5, whether low-temperature deviations distinguish nonconforming from conforming batches remains undetermined. Neither a causal association nor the validity or invalidity of a temperature-based discriminator is established by these incomplete records.

Threshold sensitivity describes only the retained production interval. Raising δ from 5 to 10 and 20 °C changes P_1 from 69.9% to 68.5% and 63.9%, and P_2 from 61.1% to 54.4% and 20.1%. These changes describe the sampled curve, not a batch-level quality relationship or independently validated quality limit. Calibration certificates and measurement-uncertainty budgets were unavailable; no numerical uncertainty or causal significance is assigned.

4. Discussion

The same-model experiment did not show a completion or stable timing advantage for the existing processing package. Its asymmetric post hoc scope review cannot support a between-arm coverage conclusion. The 41-case fallback audit identifies six whole answers supported relative to the saved evidence, but it is AI-assisted and not independently expert-validated. More fundamentally, the alarm-path defect has not been bounded across endpoints; neither returned absences nor token-level answer fidelity establish enterprise-fact correctness. Appropriate next steps include explicit temporal/identifier contracts, contemporaneous backend/tool snapshots, and independent answer assessment; their effectiveness has not been tested here.

The checked historical references cannot establish the relationship between low-temperature readings and hardness outcomes because coverage is incomplete and process conditions are not controlled. Further investigation requires comparable recipes, specimen-to-basket links, calibration records, and authorized process comparisons. A separate literature example is Li et al.'s mesh-belt study [39]. That study used DeepSeek-R1 to suggest 78 rather than 80 Hz and physically tested the recommendation; Table 4 in [39] reports an ovality reduction with an end-face-diameter trade-off. It illustrates testing of a parameter suggestion, not a new result for the present GPT–OSS system.

Interaction efficiency requires paired graphical user interface (GUI) and generative artificial intelligence (GenAI) execution records, which were not collected here. Future engineering studies could test action counts, completion times, and whether an action-distribution summary adds information beyond trace length [40,41]. Repeating a single action yields zero action-type entropy regardless of repetition count, and permuting a trace does not change that entropy; it cannot by itself measure workload or order. Incident

classification also requires separate validation: alarm management and event consequences are distinct problems [42,43].

5. Conclusions

This study investigated natural-language record access and quality traceability in a heat-treatment MES using explicit inspection, product, basket, and process-time relationships. A 48-task same-model experiment produced 273 complete responses in 288 attempts. The processing package produced fewer complete answers and no stable timing advantage. Its asymmetric post hoc scope corrections are not used as comparative evidence. Review of 41 numerical fallbacks found six pre-fallback answers supported relative to the saved evidence, 34 with unsupported or incorrect claims, and one indeterminate; these AI-assisted assessments are not independent factual validation.

Three quality cases illustrate inspection-to-process retrieval within stated record-coverage limits. Whether the retrieved temperature deviations distinguish nonconforming from conforming batches remains undetermined, and no causal relationship is established. The alarm-path defect and unverified historical absence returns further limit enterprise-fact interpretation. Future studies require contemporaneous source snapshots, symmetric blinded comparisons, and process-matched verification; user-efficiency and incident-classification claims require separate empirical evidence.

Supplementary Materials: The following supporting information can be downloaded at https://www.mdpi.com/article/10.3390/pr14213597/s1: Supplementary File S1: study design and execution settings; task definitions and reference conditions; tool catalog and common prompt; scoring and aggregate-reproduction rules; an end-to-end retrieval trace; quality-record and interval definitions; independent alarm checks; and retrospective returned-state and numerical-fallback audits. Accompanying JSON, CSV, and Python files provide the released definitions, execution metrics, audit decisions, and recalculation scripts.

Author Contributions: Conceptualization, Y.S. and W.Z.; methodology, Y.S., W.Z. and X.D.; software, Y.S.; validation, Y.S., C.L. and D.J.; formal analysis, Y.S. and X.D.; investigation, Y.S. and C.L.; resources, C.L. and D.J.; data curation, Y.S.; writing—original draft preparation, Y.S.; writing—review and editing, W.Z., X.D. and D.J.; visualization, Y.S.; supervision, W.Z. and D.J.; project administration, W.Z. and D.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Open Project Fund of the Liaoning Provincial Key Laboratory of Internet of Things Application Technology for Intelligent Construction, grant number KFKT2025-07.

Data Availability Statement: Sanitized task definitions, tool schemas, prompts, scoring rules, an end-to-end trace, and aggregate results are provided in Supplementary File S1. Original requests, deployment configuration, and enterprise records remain restricted because of commercial confidentiality. The released examples do not permit independent reconstruction of all business facts.

Acknowledgments: During preparation of this manuscript and study, the authors used OpenAI Codex for support with experimental-protocol design, benchmark and statistical scripting, retrospective record auditing, language revision, and figure generation. The authors reviewed and edited all outputs and take full responsibility for the content of this publication. The GPT–OSS 20B model evaluated in the experiment was independent of the manuscript-writing assistant.

Conflicts of Interest: Authors Chao Liu and Dongying Ju are employed by Zhejiang XCC Group Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Kusiak, A. Smart Manufacturing. *Int. J. Prod. Res.* **2018**, *56*, 508–517. [CrossRef]
2. Ma, Y.; Zheng, S.; Yang, Z.; Zheng, P.; Leng, J.; Hong, J. Leveraging large language models in next generation intelligent manufacturing: Retrospect and prospect. *J. Manuf. Syst.* **2025**, *82*, 809–840. [CrossRef]
3. Shafiee, S. Generative AI in Manufacturing: A Literature Review of Recent Applications and Future Prospects. *Procedia CIRP* **2025**, *132*, 1–6. [CrossRef]
4. Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.t.; Rocktäschel, T.; et al. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In Proceedings of the Advances in Neural Information Processing Systems, Virtual, 6–12 December 2020; Volume 33, pp. 9459–9474.
5. Chen, L.C.; Pardeshi, M.S.; Liao, Y.X.; Pai, K.C. Application of Retrieval-Augmented Generation for Interactive Industrial Knowledge Management via a Large Language Model. *Comput. Stand. Interfaces* **2025**, *94*, 103995. [CrossRef]
6. Wan, Y.; Chen, Z.; Liu, Y.; Chen, C.; Packianather, M. Empowering LLMs by Hybrid Retrieval-Augmented Generation for Domain-Centric Q&A in Smart Manufacturing. *Adv. Eng. Inform.* **2025**, *65*, 103212. [CrossRef]
7. Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; Cao, Y. ReAct: Synergizing Reasoning and Acting in Language Models. In Proceedings of the International Conference on Learning Representations, Kigali, Rwanda, 1–5 May 2023.
8. Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Hambro, E.; Zettlemoyer, L.; Cancedda, N.; Scialom, T. Toolformer: Language Models Can Teach Themselves to Use Tools. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2023; Volume 36, pp. 68539–68551. [CrossRef]
9. Choi, H.; Jeong, J. Domain-Specific Manufacturing Analytics Framework: An Integrated Architecture with Retrieval-Augmented Generation and Ollama-Based Models for Manufacturing Execution Systems Environments. *Processes* **2025**, *13*, 670. [CrossRef]
10. Yuan, Z.; Li, M.; Liu, C.; Han, F.; Huang, H.; Dai, H.N. Chat with MES: LLM-driven user interface for manipulating garment manufacturing system through natural language. *J. Manuf. Syst.* **2025**, *80*, 1093–1107. [CrossRef]
11. Fan, S.K.S.; Wang, J.T.; Jen, C.H. Integrating large language models into manufacturing execution systems: A systematic framework for photoresist manufacturing processes. *Comput. Ind. Eng.* **2026**, *211*, 111599. [CrossRef]
12. Elbasheer, M.; Laili, Y.; Longo, F.; Solina, V.; Tao, Y.; Veltri, P.; Zhang, Y.; Zhang, L. Natural Language-Driven Production Planning: Integrating Large Language Models with Automatic Simulation Model Generation in Manufacturing Systems. *J. Intell. Manuf.* **2026**, *37*, 4431–4458. [CrossRef]
13. Wu, T.; Li, J.; Bao, J.; Liu, Q. ProcessCarbonAgent: A large language models-empowered autonomous agent for decision-making in manufacturing carbon emission management. *J. Manuf. Syst.* **2024**, *76*, 429–442. [CrossRef]
14. Lim, J.; Vogel-Heuser, B.; Kovalenko, I. Large Language Model-Enabled Multi-Agent Manufacturing Systems. In Proceedings of the 2024 IEEE 20th International Conference on Automation Science and Engineering, Bari, Italy, 28 August–1 September 2024; pp. 3940–3946. [CrossRef]
15. Zhao, Z.; Tang, D.; Liu, C.; Wang, L.; Zhang, Z.; Zhu, H.; Chen, K.; Nie, Q.; Ji, Y. A Large Language Model-Based Multi-Agent Manufacturing System for Intelligent Shopfloors. *Adv. Eng. Inform.* **2026**, *69*, 103888. [CrossRef]
16. Xia, Y.; Dittler, D.; Jazdi, N.; Chen, H.; Weyrich, M. LLM Experiments with Simulation: Large Language Model Multi-Agent System for Simulation Model Parametrization in Digital Twins. In Proceedings of the 2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation, Padova, Italy, 10–13 September 2024; pp. 1–4. [CrossRef]
17. Gautam, A.; Aryal, M.R.; Deshpande, S.; Padalkar, S.; Nikolaenko, M.; Tang, M.; Anand, S. IIoT-enabled digital twin for legacy and smart factory machines with LLM integration. *J. Manuf. Syst.* **2025**, *80*, 511–523. [CrossRef]
18. Bajestani, M.S.; Mun, D.; Kim, D.B. Human-in-the-loop and large language models in smart manufacturing: Current applications, challenges, and perspectives. *J. Manuf. Syst.* **2026**, *86*, 913–941. [CrossRef]
19. Galdino, M.; Hamann, T.; Abdelrazeq, A.; Isenhardt, I. Large Language Model-Based Cognitive Assistants for Quality Management Systems in Manufacturing: A Requirement Analysis. *Eng. Rep.* **2025**, *7*, e70437. [CrossRef]
20. Zhou, B.; Li, X.; Liu, T.; Xu, K.; Liu, W.; Bao, J. CausalKGPT: Industrial structure causal knowledge-enhanced large language model for cause analysis of quality problems in aerospace product manufacturing. *Adv. Eng. Inform.* **2024**, *59*, 102333. [CrossRef]
21. Shi, D.; Liedl, P.; Bauernhansl, T. Interoperable information modelling leveraging asset administration shell and large language model for quality control toward zero defect manufacturing. *J. Manuf. Syst.* **2024**, *77*, 678–696. [CrossRef]
22. Wen, S.; Li, F.; Zhuang, W.; Pan, X.; Yu, W.; Bao, J.; Li, X. Leveraging large language models for Human-Machine collaborative troubleshooting of complex industrial equipment faults. *Adv. Eng. Inform.* **2025**, *65*, 103235. [CrossRef]
23. Li, Y.; He, Y.; Lian, R.; Guo, Q. Fault Diagnosis and System Maintenance Based on Large Language Models and Knowledge Graphs. In Proceedings of the 2023 5th International Conference on Robotics, Intelligent Control and Artificial Intelligence, Hangzhou, China, 1–3 December 2023; pp. 589–592. [CrossRef]
24. Vidyaratne, L.; Lee, X.Y.; Kumar, A.; Watanabe, T.; Farahat, A.; Gupta, C. Generating Troubleshooting Trees for Industrial Equipment using Large Language Models (LLM). In Proceedings of the 2024 IEEE International Conference on Prognostics and Health Management, Spokane, WA, USA, 17–19 June 2024; pp. 116–125. [CrossRef]

25. Pang, Z.; Zhang, H.; Li, T. Hybrid Fine-Tuning in Large Language Model Learning for Machinery Fault Diagnosis. In Proceedings of the 2024 IEEE 22nd International Conference on Industrial Informatics, Beijing, China, 18–20 August 2024; pp. 1–6. [CrossRef]
26. Zhang, Q.; Xu, C.; Li, J.; Sun, Y.; Bao, J.; Zhang, D. LLM-TSFD: An Industrial Time Series Human-in-the-Loop Fault Diagnosis Method Based on a Large Language Model. *Expert Syst. Appl.* **2025**, *264*, 125861. [CrossRef]
27. Lin, L.; Zhang, S.; Fu, S.; Liu, Y. FD-LLM: Large Language Model for Fault Diagnosis of Complex Equipment. *Adv. Eng. Inform.* **2025**, *65*, 103208. [CrossRef]
28. Li, D.; Pang, Z.; Chen, Y.; Yang, K.; Shao, J.; Luo, Y.; Zeng, Y.; He, C.; Gao, Y. FD-MVLLM: Fault Diagnosis Based on Multimodal Vibration Data and Large Language Model for Bearing System. *Mech. Syst. Signal Process.* **2025**, *239*, 113226. [CrossRef]
29. Peng, C.; Peng, J.; Wang, Z.; Wang, Z.; Chen, J.; Xuan, J.; Shi, T. Adaptive Fault Diagnosis of Railway Vehicle On-Board Controller with Large Language Models. *Appl. Soft Comput.* **2025**, *185*, 113919. [CrossRef]
30. Alsaif, K.M.; Albeshri, A.A.; Khemakhem, M.A.; Eassa, F.E. Multimodal Large Language Model-Based Fault Detection and Diagnosis in Context of Industry 4.0. *Electronics* **2024**, *13*, 4912. [CrossRef]
31. Tasneem, O.; Pieters, R. Human–Robot Collaborative Visual Inspection with Large Language Models. *Robot. Comput. Integr. Manuf.* **2026**, *98*, 103154. [CrossRef]
32. Zhang, K.; Tsang, Y.P.; Lee, C.K.M.; Wu, C.H. Integrating Large Language Models with Explainable Fuzzy Inference Systems for Trusty Steel Defect Detection. *Pattern Recognit. Lett.* **2025**, *192*, 29–35. [CrossRef]
33. Liu, S.; Koch, D.; Kaiser, J.P.; Stamer, F. Automating Industrial Quality Assurance: A Zero-Shot MLLM Framework for Defect Detection and 3R. *Procedia CIRP* **2026**, *139*, 313–318. [CrossRef]
34. Gu, S.; Lei, Z.; Wen, G.; Xu, Q.; Li, Z.S.; Chen, X.; Yang, C. WamGLM: A multimodal large-scale language model for wafer map defect information in-depth query through multi-turn dialogue based on prototypical supervised contrastive learning. *Appl. Soft Comput.* **2025**, *185*, 113962. [CrossRef]
35. Shen, X.; Li, L.; Ma, Y.; Xu, S.; Liu, J.; Yang, Z.; Shi, Y. VLCIM: A Vision-Language Cyclic Interaction Model for Industrial Defect Detection. *IEEE Trans. Instrum. Meas.* **2025**, *74*, 2538713. [CrossRef]
36. Chen, Z.; Imani, F. A Multi-Expert Framework for Enhancing Multimodal Large Language Models in Industrial Anomaly Detection. *Pattern Recognit.* **2026**, *172*, 112752. [CrossRef]
37. Ji, Z.; Lee, N.; Frieske, R.; Yu, T.; Su, D.; Xu, Y.; Ishii, E.; Bang, Y.J.; Madotto, A.; Fung, P. Survey of Hallucination in Natural Language Generation. *ACM Comput. Surv.* **2023**, *55*, 1–38. [CrossRef]
38. Efron, B. Bootstrap Methods: Another Look at the Jackknife. *Ann. Stat.* **1979**, *7*, 1–26. [CrossRef]
39. Li, X.; Sun, Y.; Deng, X.; Wang, J.; Ju, Y.; Wang, M.; Wang, L.; Chen, H.; Ju, D. Application of AI-Driven Methods in Quenching Distortion Control of Mesh Belt Furnaces. *Processes* **2026**, *14*, 718. [CrossRef]
40. Shannon, C.E. A Mathematical Theory of Communication. *Bell Syst. Tech. J.* **1948**, *27*, 379–423. [CrossRef]
41. Card, S.K.; Moran, T.P.; Newell, A. The Keystroke-Level Model for User Performance Time with Interactive Systems. *Commun. ACM* **1980**, *23*, 396–410. [CrossRef]
42. Goel, P.; Datta, A.; Mannan, M.S. Industrial Alarm Systems: Challenges and Opportunities. *J. Loss Prev. Process. Ind.* **2017**, *50*, 23–36. [CrossRef]
43. Mustafa, F.E.; Ahmed, I.; Basit, A.; Alvi, U.E.H.; Malik, S.H.; Mahmood, A.; Ali, P.R. A Review on Effective Alarm Management Systems for Industrial Process Control: Barriers and Opportunities. *Int. J. Crit. Infrastruct. Prot.* **2023**, *41*, 100599. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.