From Qualitative Roles to Measurable Evidence: Classical Text Representations for Action-Type Recognition in Manufacturing Execution Systems

Surya T. Meesala

***Abstract*— Manufacturing execution systems generate large volumes of informal knowledge, including shift log books, work instructions, and maintenance logs. However, concrete, quantitative evidence of the use of generative artificial intelligence (GenAI) in the shop-floor is rare. We present an empirical study on a publicly available expert-annotated maintenance short text dataset (MaintIE). This dataset comprises 1,076 records, where the task is to predict the maintenance action type (Replace, Repair, Inspect, Service, Diagnose, Adjust, Calibrate) from a free-text work-order description. We evaluate classical text vector representations combined with linear classifiers (stratified five-fold cross-validation). The representations and classifiers are deliberately standard; the contribution is instrumental rather than algorithmic, namely a quantitative characterization of where such methods suffice on MES text and where their applicability ends. Testing shows that the character n-gram representation (with logistic regression) achieves a macro-F1 score of 0.889 and accuracy of 0.956 while the baseline, majority class predictor, achieves a macro-F1 score of 0.107. A noise-injection experiment that simulates typographic errors and common abbreviations shows that character n-grams degrade by 8% at 50% token corruption while words degrade by 38%. Two neural baselines evaluated under the same protocol place these numbers on a common scale: frozen general-purpose sentence embeddings match the character n-gram model to within 0.002 macro-F1, and a fine-tuned transformer gains 0.043 at an order of magnitude more training cost. A similarity-based retrieval experiment estimates the coverage-precision trade-off that bounds the case range where lexical retrieval is sufficient. A comparison of retrieval for the two classes suggests that mid-weight semantic retrieval is on average more precise on the remaining class by about 10 precision points. This transposes the classical vs. semantic boundary from a presupposition into a quantifiable measurement. This work converts qualitative statements about MES knowledge work into reproducible measurements and presents an evidence-based, graduated deployment roadmap.**

***Keywords*—maintenance work orders, manufacturing execution systems, technical language processing, text classification.**

I. INTRODUCTION

The Manufacturing Execution System (MES), the control layer of digital manufacturing, closes the divide between enterprise management and physical production, controlling operation sequencing, batch and product state, and material and parameter traceability and quality reporting at shift and line granularity [1]. As production moves towards shorter batch runs, and increased variation, the number of process-variant deviations becomes larger and more heterogeneous. Much of the knowledge required to interpret the deviations is not recorded in discrete fields, but in unstructured text such as shift logs, work instructions, investigation reports and change management emails [2, 3]. This has been described as an underused source of information not being used in decision making, motivating the need for dedicated technical language processing [4].

Several recent papers propose generative AI/LLMs to unlock this knowledge. Notable works include generative conversational assistants, auto-summarization of shifts, and using RAG on corporate knowledge bases [5-7]. Writing on these technologies has tended to be qualitative. Given that these are architectural taxonomies and roadmaps that do not quantify the shop floor data, it is difficult to judge when auditing methods are cheaper than the costs and risks of the generative models.

Rather than trying to provide direct evidence that generative artificial intelligence is useful for MES, the paper focuses on the specific question of whether generative artificial intelligence can help with public datasets. The first aspect is whether work-type classification can be performed on a raw work-order description with the help of classical text encoders, and whether the work-type classification ability of encoders is affected by the noisy and abbreviated language used in real maintenance text. Second, what are the limitations of similarity-based retrieval, the most common backbone of RAG pipelines?

The novelty of this paper is instrumental rather than algorithmic. Character n-grams, TF-IDF weighting and linear classifiers are long-established techniques, and we make no claim to improve on them; what has been missing is a quantitative account of how far they carry on MES text and of the point at which heavier machinery has to earn its cost. This paper makes four contributions. First, it establishes a reproducible reference point for action-type recognition on the public MaintIE corpus [8], formulated as an operationally meaningful MES classification task and measured with deliberately standard representations and linear classifiers. Second, it measures, through controlled noise injection, the robustness advantage of character-level representations over word-level representations on degraded shop-floor text. Third, it measures the coverage-precision trade-off in lexical retrieval and compares lexical, semantic, and hybrid retrieval

[1]Manuscript received June 5, 2026.

S. T. Meesala is an Independent Researcher from Memphis, TN, USA (e-mail: surya7ms@gmail.com).

on the residual segment left uncovered by lexical retrieval, thereby making the reach of a semantic representation a measurable quantity rather than an assumption. Fourth, from these measurements, it derives a staged, read-only-first deployment roadmap for generative artificial intelligence overlays on MES.

II. Related Work

A. MES and Non-Formalized Production Knowledge

Intelligent MES has been described as an arbiter for equipment, personnel, and higher-level software systems to synchronize across multiple types of data and provide real-time views of progress of work [1]. The increasing number of variants and lack of skilled labor provide push for digital decision support on the shop floor [2, 3]. This phenomenon of capturing operational knowledge in unstructured technical text is a recurrent theme in this work, motivating the use of natural language processing (NLP) in the maintenance domain [4].

B. Technical Language Processing for Maintenance

However, short texts produced by maintenance technicians are often not only very short, but also contain abundant abbreviations and technical terminology that are not well suited for general-purpose NLP pipelines [4, 8]. The publicly available MaintNet [4] and the more granular MaintIE schema and corpus [8] allow to benchmark different systems. Studies on wind-turbine maintenance logs provided comparisons among human-to-human, supervised classification, and AI labeling of data using key performance indicator extraction and validation [9, 10]. A recent comparative evaluation bench-marks open-source LLMs in labeling logs and compares the cost of computation for detecting hallucination [11]. Reviews have surveyed natural language processing in domain-specific applications and call it a move from classical classification of terms to LLM-based extraction, with restrictions imposed by privacy and cost constraints [12].

C. Generative AI and Retrieval in Manufacturing

Generative methods can also be integrated with existing MES functions, such as conversational agents, analytics, and knowledge bases [5, 6]. RAG connects model outputs to company knowledge to improve verifiability and reduce hallucinated output [7, 13]. Further, industrial RAG studies in manufacturing quality control [14] and industrial knowledge representation for cognitive digital twins [15] show that even the best embedding models do not rank the gold candidate as the top item when the distractor candidates are semantically similar to it. This indicates that retrieval, not generation, performance constrains the end-to-end performance of RAG, motivating our focus on retrieving top-k instances.

III. Materials and Methods

The MaintIE corpus [8] (distributed under MIT license) features coarse- and fine-grained annotation for 1,076 maintenance-related texts. It provides fine-grained annotation for 1,076 maintenance-related texts and coarse-grained annotation for 7000 texts. Each fine-grained entry is multi-stage sub-blocked with a five-class entity annotation scheme consisting of PhysicalObject, State, Process, Activity, and Property. Text lengths vary between 2 and 12 tokens, and the average of around 5.7 tokens per message is typical of the abbreviations and non-standard spellings such work orders are known to use.

For the action-type recognition sub-task, we also take the dominant maintenance activity from Activity and MaintenanceActivity entities as a single document-level label for each record, without ambiguity. Indeed, we ignore any record without any maintenance-activity entity and any record that has more than two types of activity in the same document (for example, in a tie). This leaves us with 564 labeled records, covering seven classes, which are imbalanced in a way that realistic maintenance data would be (see Table 1). The task mirrors a realistic MES demand by inferring the type of intervention described in the free text. This inference is relevant for dispatching, planning, and reporting.

Table 1. Action-type label distribution across 564 records

Action type	Count	Share
Replace	338	59.9%
Repair	108	19.1%
Inspect	49	8.7%
Service	33	5.9%
Diagnose	17	3.0%
Adjust	10	1.8%
Calibrate	9	1.6%
Total	**564**	**100%**

The first representation is a word-level TF-IDF representation computed over unigrams and bigrams, sublinear scaling of term frequency, and a minimum document frequency of two. The second representation is a character-level TF-IDF representation computed as above but instead over within-word n-grams of lengths two, three, four, and five. The intuition is that character n-grams are more strong against the noise and abbreviation patterns present in maintenance text, since they share sub-word evidence across surface-level manifestations. For each representation, we apply three linear classifiers: logistic regression, a linear support vector classifier (both with class-balanced weights to address data imbalance), and a Complement Naive Bayes classifier, which is often employed in imbalanced text classification. The baseline most-frequent-class predictor is used for comparison.

All models are trained and evaluated using stratified five-fold cross-validation, with macro-averaged F1 the primary metric because it equally weighs each action type and is a more useful measure given imbalanced operational data. Accuracy is a secondary metric. Per-class F1 and normalized confusion matrix are obtained on cross-validated results. The ranges correspond to the standard deviation across folds. The entire pipeline runs under scikit-learn [16] and is deterministic for a fixed random seed.

To place these classical results on a common scale with contemporary neural methods, we additionally evaluate a

lightweight transformer baseline under an identical protocol: the same 564 records, the same stratified five-fold splits, the same macro-averaged F1 as primary metric and the same fixed random seed. Two variants are reported. The first uses frozen MiniLM sentence embeddings as input to the same logistic-regression head; the second fine-tunes DistilBERT end-to-end. The fine-tuned variant is trained for six epochs with AdamW at a learning rate of 3e-5, a batch size of 16, a maximum sequence length of 32 tokens and the same balanced class weighting as the linear models. The purpose is not to select a winner but to quantify the gap between an interpretable bag-of-n-grams pipeline that runs on commodity hardware and a neural model that requires pre-trained weights and, in the fine-tuned case, gradient updates. That gap has a price: the full classical pipeline completes in about two minutes on a commodity CPU, whereas fine-tuning takes twenty-five to forty minutes on the same machine.

In this robustness test, noise is added to the test data, but the model is trained on the clean data, approximating the situation when the incoming data has some noise, while the model is trained on only the clean texts of curated historical data. The operations are transposing neighboring characters, deleting and duplicating characters, and replacing common domain-specific abbreviations, like repl with replace and brg with bearing. Each is applied at a given frequency per token. The corruption rate ranges from 0% to 50%.

In the similarity-based retrieval experiment, the training fold is treated as a knowledge base, and each record in the testing fold is treated as a query. Using the character n-gram representation, the most similar previously seen records are determined using cosine similarity. The evaluation metrics are the precision at k value (that is, whether the retrieved records have the same type of action as the query record), the mean reciprocal rank and the traceability rate, which is defined as the fraction of queries for which at least one source is retrieved above a defined similarity threshold. A threshold sweep shows the trade-off between coverage and precision of accepted sources.

Another test verifies that a semantic method closes the remaining segment left by the lexical lookup (i.e. the remainder). Queries from each fold are split into two sets. The easy subset contains queries with lexical top-1 similarity scores above or equal to a high confidence threshold, and the residual subset contains queries with those below this threshold. Five retrievers are compared on each subset. The first is the lexical char n-gram cosine. The second is a semantic retriever based on latent semantic analysis, where we apply truncated singular-value decomposition to the TF-IDF matrix and compute the cosine similarity in this latent space. The third is a hybrid which averages the previous two similarities. The fourth is a neural retriever built on frozen MiniLM sentence embeddings, and the fifth averages the lexical and neural similarities. Latent semantic analysis is a middle-ground semantic model that runs on commodity hardware and needs no pre-trained neural weights, which makes it a reproducible reference point for the gains a semantic representation can provide. Whether a neural encoder exceeds that reference point is treated here as a question to be measured rather than assumed, which is why the two neural retrievers are evaluated alongside it. The number of latent dimensions can also vary.

IV. RESULTS

A. *MES and Non-Formalized Production Knowledge*

Table 2 shows the cross-validated performance of all six of the representation/classifier combinations. The majority-class classifier has an accuracy of 0.599 and macro-F1 of 0.107, showing how misleading accuracy can be with highly imbalanced data. The best-performing model was character n-grams with logistic regression (macro-F1: 0.889, accuracy: 0.956). Sub-word representations outperformed word representations for all classifiers, with the best character model improving on the best word model by 0.056 macro-F1. This improvement shows the value of sub-word evidence, especially on clean text.

Two neural baselines evaluated under the same protocol place these figures on a common scale. On splits re-derived from the corpus, on which the character n-gram model scores 0.877 macro-F1, frozen MiniLM sentence embeddings fed to the same logistic-regression head reach 0.875 ± 0.057, that is, no measurable difference (paired Wilcoxon across folds, $\Delta = 0.002$, $p = 1.0$). A fine-tuned DistilBERT reaches 0.920 ± 0.055, an advantage of 0.043 macro-F1 that a paired test on five folds cannot separate from zero ($p = 0.31$; with five folds the smallest attainable p-value is 0.0625, so this design can indicate but not establish significance). The practical reading is that a general-purpose sentence encoder buys nothing over character n-grams on this task, and that end-to-end fine-tuning buys a few points of macro-F1 in exchange for pre-trained weights, gradient updates and the feature-level interpretability shown in Figs. 5 and 6.

Table 2. Five-fold cross-validation results

Representation	Classifier	Macro-F1	Accuracy
Majority baseline	-	0.107	0.599
Word 1–2-gram	Logistic regression	0.833 ± 0.066	0.927 ± 0.023
Word 1–2-gram	Linear SVC	0.824 ± 0.062	0.938 ± 0.020
Word 1–2-gram	Complement NB	0.746 ± 0.047	0.895 ± 0.014
Char 2–5-gram	**Logistic regression**	**0.889 ± 0.057**	**0.956 ± 0.021**
Char 2–5-gram	Linear SVC	0.878 ± 0.063	0.957 ± 0.017
Char 2–5-gram	Complement NB	0.841 ± 0.059	0.929 ± 0.015
MiniLM embeddings	Logistic regression	0.875 ± 0.057	0.936 ± 0.010
DistilBERT	Fine-tuned	0.920 ± 0.055	0.977 ± 0.015

The word- and character-level rows are taken from the original run. The two neural rows were measured on splits re-derived from the corpus, on which the character n-gram model scores 0.877 ± 0.071 macro-F1 and 0.947 ± 0.028 accuracy; the neural comparisons drawn in the text are made against those figures.

Fig. 1 shows per-class behavior. High frequency, lexically distinct classes (Replace, Repair, Inspect) achieve F1 scores above 0.94. The rare Calibrate class however has 100% recovery because its vocabulary distinguishes it

lexically from the other classes.

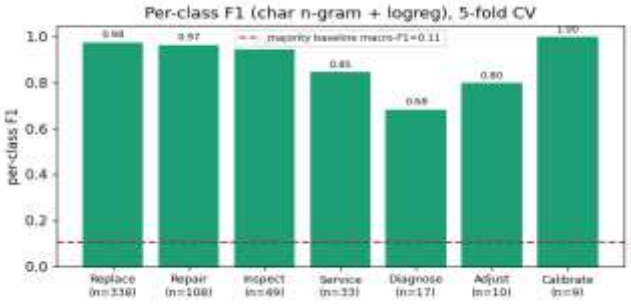

Fig. 1. Per-class F1 of the best model, character n-grams with logistic regression, under five-fold cross-validation

The class with the worst F1-score was the rare and lexically diffuse Diagnose class with 0.685. The normalized confusion matrix in Fig. 2 shows that the remaining errors are due to confusion with the majority Replace class, not because of systematic leakage across classes.

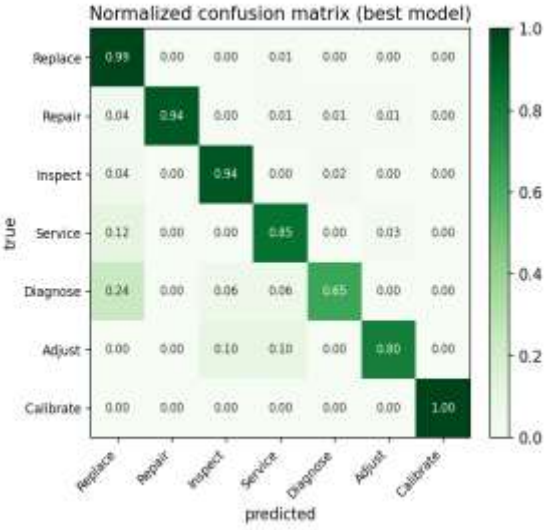

Fig. 2. Row-normalized confusion matrix of the best model from cross-validated predictions

The learning curve in Fig. 3 shows that performance rapidly improves until about 300 training records, after which performance starts to plateau. Performance on the training set and validation set converge to the error of the rarest classes. This shows that even a small curated corpus is useful, and that rare action types would benefit the most from additional labeled examples.

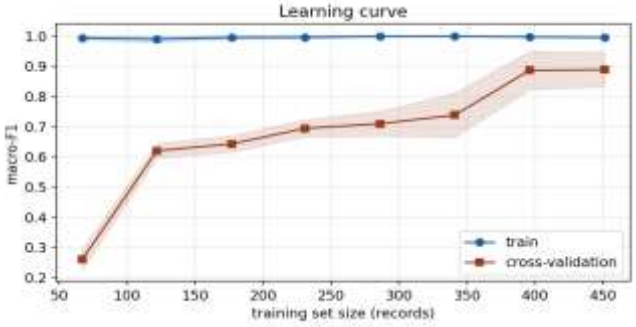

Fig. 3. Learning curve, macro-F1 against training-set size, for the best model

A 2D t-SNE projection of the character n-gram representation from Fig. 4 qualitatively supports these properties. Calibrate and Inspect are separated and compact, while Diagnose is more spread out and interspersed with the other classes, which matches the finding from the F1 measure that the classifier struggles to perform well on Diagnose.

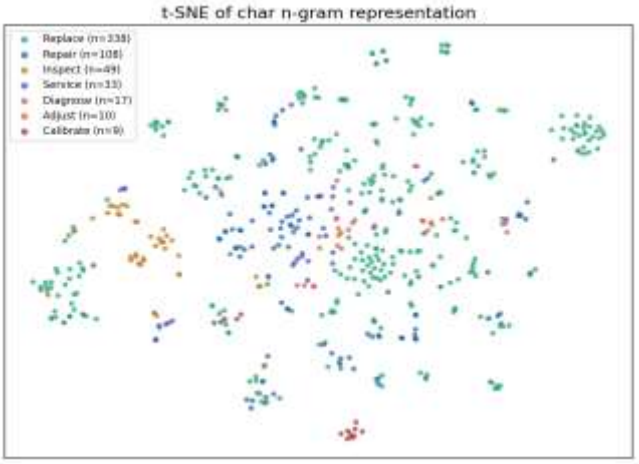

Fig. 4. t-SNE projection of the character n-gram representation, colored by action type

Figures 5 (most discriminative character n-grams per class) also show that the model is able to rely on interpretable sub-word features, such as combinations of characters "repl", "nspe" and "serv", separated by \u25A1, which resembles an open box. This is particularly important for the case of MES, where all automated suggestions need to be explainable and auditable.

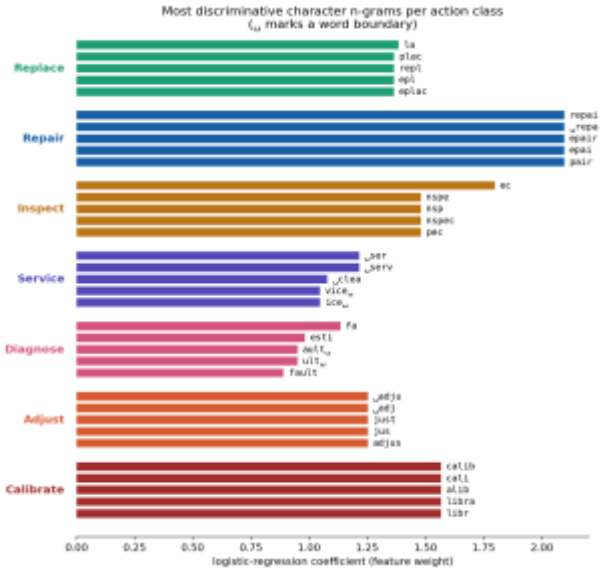

Fig. 5. Top five most discriminative character n-grams per action class, ranked by logistic-regression coefficient

The same evidence can be visualized in the form of a class-by-feature weight matrix (Fig. 6). Here, the matrix is almost block-diagonal as the models use largely disjoint character n-gram sets for each action type. This explains the strong class separability and the concentrated errors for the lexically diffuse Diagnose class.

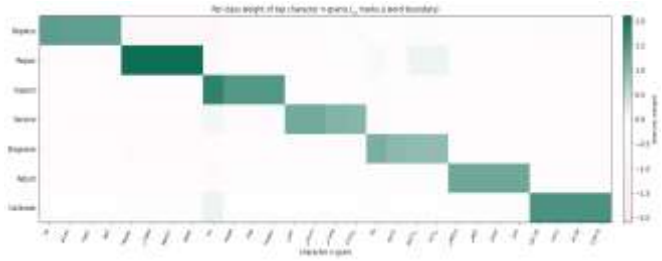

Fig. 6. Per-class weight of the leading character n-grams, which shows a near block-diagonal structure

B. *Robustness to Text Noise*

In Fig. 7, we plot the macro-F1 score against injected noise. For clean text inputs, the character model is initially winning. As corruption increases, this gap also increases, from 0.056 when there is no noise, to 0.302 when there is 50% noise. When the input data has 50% corruption, the word-level model's macro-F1 score drops by 38%, from 0.833 to 0.514, but the character-level model's macro-F1 score drops by only 8%, from 0.889 to 0.816. For MES scenarios where the work order text is often entered by operators using fast inputs, without strict formatting, this robustness supports sub-word representations.

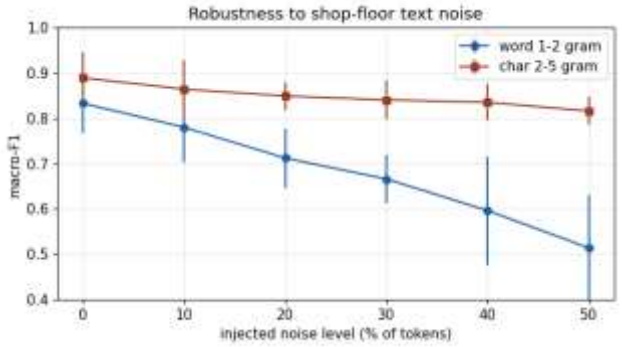

Fig. 7. Robustness to shop-floor text noise, macro-F1 against injected token corruption for word and character representations

C. *Retrieval of Similar Past Cases*

Using the historical fold as the corpus, similarity-based retrieval retrieves a same-action past case with precision 0.794 (compared with 0.407 by chance) and mean reciprocal rank 0.861. A lightweight lexical retriever already gives a useful way to correlate symptoms with failure history, and it gives an explicit, citable source for each past case, which is the prerequisite for auditability.

The quality of their retrieval is shown together with their acceptance threshold, which is described by the trade-off in Fig 8. For the permissive threshold, every query has a source, but the precision of accepted sources is only 0.79. If the precision of accepted sources is 0.93, then the threshold is higher and the coverage is lower, at 43%. This suggests that a lexical retriever will not produce a high-confidence source for 57% of queries, which is the point at which we switch to using semantic embeddings or generative reasoning. This is in agreement with industrial evidence showing that retrieval quality, and not just generation quality, is key to end-to-end performance [15].

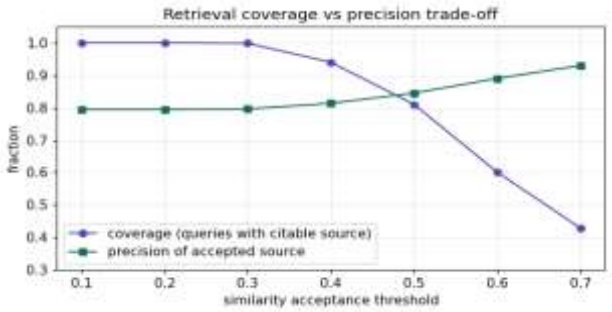

Fig. 8. Retrieval coverage against precision of accepted sources as a function of the similarity acceptance threshold

D. *Does a Semantic Method Close the Residual Segment?*

Although the above coverage analysis identifies a residual segment, it is not clear whether this segment can be recovered by a semantic representation. This issue is settled by a segmented experiment. With the high-confidence threshold, the residual segment comprises 57% of queries, as in Section IV-C experiments. The three retrievers are noisily tied on the easy queries with lexical, semantic, and hybrid scores of 0.930, 0.930, and 0.938, respectively. However, in these contexts the surface lexical evidence is so strong that there is no potential for the semantics to contribute. For the remaining data, top-1 precision increases from 0.693 to 0.796 (absolute gain of 10.3) if the latent-semantic retriever is employed, and to 0.768 if the latent-semantic hybrid is leveraged, showing that a part of the residual has a semantic structure (see Fig. 9a).

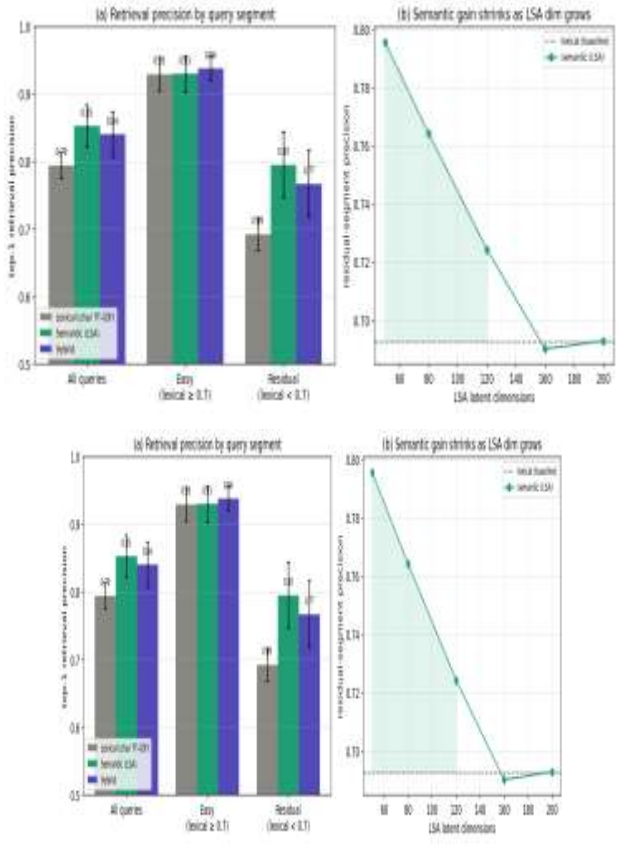

Fig. 9. Retrieval precision by query segment and semantic gain shrinks as LSA dim grows

Whether the gain is larger or smaller thus depends on the

latent dimensionality. As illustrated in Fig. 9b, the gain is largest for the low dimensionalities, where the decomposition compresses surface features onto a few semantic axes capturing the synonymy and co-occurrence patterns, and shrinks for larger dimensionalities when the latent space approximates the original lexical space. This holds for all acceptance thresholds, with a 2-3 point gain for high dimensionality and up to 10 points for low dimensionality, and is reproducible across thresholds. Such a mid-weight semantic method therefore recovers a measurable part of the residual, and how much a neural embedding model adds on top of it is no longer left as an assumption, but measured directly below.

A general-purpose neural retriever was added to the same segmented comparison. On re-derived splits, on which lexical top-1 precision over the residual is 0.683 and the latent-semantic retriever reaches 0.787, frozen MiniLM sentence embeddings reach only 0.667, below the lexical baseline itself, and a lexical-neural hybrid reaches 0.699; the residual segment covers 55.3% of queries under these splits. Off-the-shelf neural embeddings therefore do not extend the latent-semantic result on this corpus, and lose ground on precisely the queries where lexical overlap is weakest. This is a statement about one general-purpose sentence encoder applied to normalized maintenance short texts containing placeholder tokens, not about domain-adapted neural retrieval, which is not tested here and remains the natural next step.

V. Discussion

These numbers support a number of claims in the MES literature that are made qualitatively: Recovering operational meaning (what action is needed) from free-text work orders can be done with transparent, inexpensive methods on commodity hardware. The good performance (macro-F1 close to 0.89) does not require a large model. The choice of representation matters, since sub-word features provide a large, measured robustness advantage in the regime where real shop-floor text is hardest, and that regime is the one which matters in operation. Third, the retrieval that lies beyond a RAG pipeline's upper bound can be measured, rather than simply assumed. A mid-weight semantic retriever does recover a measurable part of the residual (roughly ten precision points), confirming that the residual is at least partially semantic. It does not, however, act as a floor beneath heavier machinery: a general-purpose neural sentence encoder scored below both the latent-semantic retriever and the lexical baseline on the same segment. The expectation that neural retrieval would extend the semantic gain does not hold on this corpus, and is therefore something to be established rather than presumed.

This lends support to an overlay approach, in which classical techniques deal reliably with the bulk of well-defined high volume cases, with generative elements best suited to tackle the more intractable residual cases, thus adding value in a controlled fashion, rather than risk-in-bulk, and thus providing a more strong basis for investment and risk management than a blanket recommendation to adopt generative artificial intelligence.

The first stage of the Evidence-based deployment roadmap is read-only deployment of the evidence. The various ML assisting components include classifications and lexical retrieval on shift summaries, action-type tagging, past cases matching, and tracing all outputs back to the source record. Production risk remains low, as no commands are yet given to the equipment.

In the second stage, the method passes ranked further suggestions and drafts to the high-confidence majority of the classifier and the high-coverage part of the retrieval curve, while always remaining in the human confirmation loop.

The third stage is selective semantic and generative augmentation. The low-coverage residual (around 57% of top-40 examples at high precision thresholds) is routed through the semantic retriever since a light latent semantic approach gives around 10 precision points on the residual with little cost. The neural embedding or generative reasoning is only for the part of the residual after semantic compression, where the cost of the embedding and proof effort is worthwhile.

Stage four allows a limited number of logged, reversible activities (such as issuing work packages and drafting spare-parts requests) upon role-based authorization, and also seeks to re-evaluate models and reference data as an active process to avoid context drift.

The limitations of the study include a single source of public corpus that is dominated by a single industrial domain and consists of short texts, limiting the extrapolation of absolute scores across different domains. The labeling scheme and dominant-action rule are simplifications. The noise model is synthetic in a sense, but similar to empirically documented maintenance-text phenomena. The comparison focuses entirely on classical or mid-range semantic methods, and it is implemented and reproduced using off-the-shelf computational hardware. These qualities deliver solidity and reproducibility. Because the residual segment comes with an upper anchor from latent-semantic analysis, the next task of benchmarking neural sentence encoders and generative extraction models to it closes a gap, rather than starting a search for an even better baseline.

VI. Conclusion

This study replaced qualitative assertions about generative artificial intelligence in MES with reproducible measurements on public maintenance text. A transparent character n-gram classifier achieves macro-F1 of 0.889 in action-type recognition. Sub-word representations hold up far better than word-level ones under realistic text noise. Neural baselines run under the same protocol bound the gap: a frozen general-purpose sentence encoder matches the character n-gram classifier to within 0.002 macro-F1, a fine-tuned transformer gains 0.043 that a five-fold paired test cannot separate from zero, and on the residual retrieval segment the same encoder falls below both the latent-semantic retriever and the lexical baseline. Lexical retrieval has a measurable coverage and precision ceiling, and a segmented comparison shows that a mid-weight semantic retriever recovers a measurable part of the residual beyond that ceiling, anchoring in numbers the point at which heavier neural or generative methods would have to earn their cost. Together, these results yield an evidence-based, staged roadmap that starts with

auditable read-only capabilities and adds semantic and generative augmentation where it adds measured value. The corpus is public under a permissive license, and the protocol is specified in full, which supports direct extension to richer models.

REFERENCES

[1] A. Shojaeinasab et al., "Intelligent manufacturing execution systems: A systematic review," Journal of Manufacturing Systems, vol. 62, pp. 503–522, 2022, doi: 10.1016/j.jmsy.2022.01.004.

[2] A. Kusiak, "Generative artificial intelligence in smart manufacturing," Journal of Intelligent Manufacturing, vol. 36, no. 1, pp. 1–3, 2025, doi: 10.1007/s10845-024-02480-6.

[3] L. Li, "Reskilling and upskilling the future-ready workforce for Industry 4.0 and beyond," Information Systems Frontiers, vol. 26, pp. 1697–1712, 2024, doi: 10.1007/s10796-022-10308-y.

[4] F. Akhbardeh, T. Desell, and M. Zampieri, "MaintNet: A collaborative open-source library for predictive maintenance language resources," in Proc. 28th Int. Conf. Computational Linguistics: System Demonstrations, 2020, pp. 7–11, doi: 10.18653/v1/2020.coling-demos.2.

[5] M. Ghobakhloo et al., "Generative artificial intelligence in manufacturing: opportunities for actualizing Industry 5.0 sustainability goals," Journal of Manufacturing Technology Management, vol. 35, no. 9, pp. 94–121, 2024, doi: 10.1108/JMTM-12-2023-0530.

[6] S. S. Sengar, A. B. Hasan, S. Kumar, and F. Carroll, "Generative artificial intelligence: a systematic review and applications," Multimedia Tools and Applications, vol. 84, pp. 23661–23700, 2025, doi: 10.1007/s11042-024-20016-1.

[7] M. Klesel and H. F. Wittmann, "Retrieval-augmented generation (RAG)," Business & Information Systems Engineering, vol. 67, pp. 551–561, 2025, doi: 10.1007/s12599-025-00945-3.

[8] T. K. Bikaun, T. French, M. Stewart, W. Liu, and M. Hodkiewicz, "MaintIE: A fine-grained annotation schema and benchmark for information extraction from maintenance short texts," in Proc. LREC-COLING, 2024, pp. 10918–10932.

[9] M.-A. Lutz et al., "KPI extraction from maintenance work orders: a comparison of expert labeling, text classification and AI-assisted tagging for computing failure rates of wind turbines," Energies, vol. 16, no. 24, art. 7937, 2023, doi: 10.3390/en16247937.

[10] J. Walgern et al., "Impact of using text classifiers for standardising maintenance data of wind turbines on reliability calculations," IET Renewable Power Generation, vol. 18, no. 15, pp. 3463–3479, 2024, doi: 10.1049/rpg2.13151.

[11] M. Malyi, J. Shek, A. McDonald, and A. Biscaya, "A comparative benchmark of large language models for labelling wind turbine maintenance logs," IET Renewable Power Generation, 2026, doi: 10.1049/rpg2.70212; preprint arXiv:2509.06813, 2025.

[12] M. Dawson, S. Lukens, and M. Sharp, "Advancing technical language processing and large language models in industrial applications," NIST Special Publication 1500-37, Dec. 2025, doi: 10.6028/NIST.SP.1500-37.

[13] P. Zhao et al., "Retrieval-augmented generation for AI-generated content: a survey," Data Science and Engineering, vol. 11, pp. 1–29, 2026, doi: 10.1007/s41019-025-00335-5.

[14] J. A. Heredia Álvaro and J. González Barreda, "An advanced retrieval-augmented generation system for manufacturing quality control," Advanced Engineering Informatics, vol. 64, art. 103007, 2025, doi: 10.1016/j.aei.2024.103007.

[15] D. Shi, J. Li, O. Meyer, and T. Bauernhansl, "Enhancing retrieval-augmented generation for interoperable industrial knowledge representation and inference toward cognitive digital twins," Computers in Industry, vol. 171, art. 104330, 2025, doi: 10.1016/j.compind.2025.104330.

[16] F. Pedregosa et al., "Scikit-learn: machine learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.