Digital Chemical Engineering 20 (2026) 100337

Contents lists available at ScienceDirect

Digital Chemical Engineering

journal homepage: www.elsevier.com/locate/dche

Original Article

Physics-structured graph neural networks for interpretable fault detection in semiconductor manufacturing processes

Chun-Pei Lin [a], Feiyang Ou [a], Yifei Wang [a], Chao Zhang [c], Sthitie Bom [c], James F. Davis [a], Panagiotis D. Christofides [a,b,*]

[a] Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA, 90095-1592, USA
[b] Department of Electrical and Computer Engineering, University of California, Los Angeles, CA 90095-1592, USA
[c] Seagate Technology, Bloomington, MN 55435, USA

ARTICLE INFO

Keywords:
Smart manufacturing
Industry 4.0
Machine learning
Virtual metrology
Soft sensing
Data engineering
Semiconductor manufacturing

ABSTRACT

Smart manufacturing has transformed semiconductor fabrication into a highly data-driven environment where large numbers of in-situ sensors continuously generate process monitoring data across multiple production tools. Leveraging these data streams to support yield improvement and fault detection has become an important challenge, particularly because conventional physical metrology is often costly, time-consuming, or destructive. Machine learning-based soft sensing has therefore emerged as a practical alternative for predicting process outcomes using routine sensor measurements and contextual process information. Previous studies have demonstrated that classical machine learning models such as gradient boosting decision trees (GBDTs) and feedforward neural networks (FNN) can improve pass/fail classification performance when applied to individual tools or aggregated intra-tool data (i.e., tools of the same class). In this work, we extend these approaches to a broader range of semiconductor manufacturing processes (i.e., data from tools belonging in different classes as well data from the same tool when making different products), and propose a physics-structured graph neural network (GNN) framework for PASS/FAIL prediction. In addition, modern tabular deep-learning models, including TabM and TabR, are evaluated as complementary benchmarks to determine whether strong neural tabular learning can improve fault-detection performance beyond conventional GBDT models and physics-structured graph learning. Instead of modeling individual sensors directly, sensor measurements are organized into physically meaningful process subsystems, each representing a group of related sensors describing process dynamics. These subsystem representations are combined with categorical contextual variables and processed through a graph attention architecture to capture interactions among process modules. TabM is used as a parameter-efficient ensemble-style neural tabular model, while TabR introduces retrieval-augmented prediction by comparing each production run with similar historical training runs. The models are evaluated using production data collected over an 18-month period. For each tool, the most recent 20% of the timeline is reserved for testing while the earlier data are used for training and validation. To ensure statistical robustness, only tools with more than 8000 observations within the study period are included. Results demonstrate that the proposed physics-structured GNN improves predictive performance on several tools compared with conventional machine learning approaches while also providing interpretable attention weights that highlight the relative contribution of different process subsystems to predicted failures. The TabM and TabR results further demonstrate that modern tabular deep-learning models can be useful complementary predictors, especially on tools with richer failure information, although they do not consistently replace the optimized LightGBM benchmark under severe class imbalance. These results demonstrate the potential of physics-guided graph learning and complementary tabular deep-learning methods for scalable and interpretable fault detection in semiconductor manufacturing environments.

* Corresponding author at: Department of Chemical and Biomolecular Engineering, University of California, Los Angeles, CA, 90095-1592, USA.
E-mail address: pdc@seas.ucla.edu (P.D. Christofides).

https://doi.org/10.1016/j.dche.2026.100337
Received 6 July 2026; Received in revised form 31 July 2026; Accepted 14 August 2026
Available online 24 August 2026

1. Introduction

The semiconductor industry is undergoing a rapid digital transformation, driven by rising microelectronics demand, increasing product and process complexity, and the financial necessity to maximize yield without escalating equipment or labor costs. In the broader context of Industry 4.0 and smart manufacturing, modern fabrication facilities deploy extensive networks of in-situ sensors and information systems across toolsets, enabling large-scale data collection that can support monitoring, diagnosis, control, and optimization (Christofides et al., 2007; Davis et al., 2020; Zhang et al., 2021). However, converting high-volume, heterogeneous equipment data into timely and reliable operational decisions remains challenging in practice, particularly when process behavior evolves over time and when production data exhibits missing values, noise, and strong tool-to-tool variability.

A central application of data-driven manufacturing intelligence is machine learning-based soft sensing, also referred to as virtual metrology, in which product quality or process outcomes are predicted from routinely collected tool-state variables and contextual process information rather than physically measured on every unit (Kadlec et al., 2009; Dreyfus et al., 2022; Maitra et al., 2024; Suthar et al., 2019). In semiconductor fabs, such predictive capability is attractive because physical metrology can be expensive, time-consuming, and sometimes destructive, and because measurement capacity can become a throughput bottleneck. Recent surveys highlight both the breadth of virtual metrology research and the persistent deployment barriers associated with model robustness under drifting conditions, limited failure rates, and the need for engineering trust and maintainability (Sun and Ge, 2021; Maitra et al., 2024). Prior industrial studies have also demonstrated that strong baseline learners such as gradient-boosted decision trees (GBDTs) and relatively simple feedforward neural network models (FNN) can achieve competitive PASS/FAIL virtual metrology performance when trained with carefully engineered features and appropriate data aggregation strategies (Ou et al., 2024, 2025).

In practical manufacturing operations, process monitoring (identifying normal versus abnormal runs) and fault diagnosis (localizing root causes) are often treated as distinct operational problems. Standard data-driven approaches typically optimize purely for the monitoring task, mapping a flattened vector of sensors to a binary $PASS/FAIL$ decision. While efficient for screening, this treats the machine as a black box and fails to provide actionable diagnostic localization.

To bridge this gap, this work leverages the monitoring task as a structured entry point for diagnosis. Rather than treating $PASS/FAIL$ prediction as an unstructured classification problem, we divide the monitoring function by the tool's underlying physical subsystems. By embedding prior structural knowledge directly into the learning system, the network must optimize its $PASS/FAIL$ prediction through the lens of coordinated subsystem dynamics. Consequently, the $PASS/FAIL$ benchmark serves as the screening mechanism, while the internal physics-structured attention layers simultaneously solve the distinct problem of diagnostic localization—translating a failure alert directly into an actionable engineering root cause.

Despite these advances, there remains a gap between high-performing predictive models and models that directly support fault analysis in real manufacturing operations. Many data-driven approaches treat the sensor table as an unstructured vector, which can obscure the physical meaning of learned correlations and make it difficult for engineers to localize likely root causes to actionable equipment subsystems. At the same time, physics-guided machine learning has increasingly been advocated as a way to improve data efficiency, interpretability, and plausibility by embedding prior physical or structural knowledge into learning systems (Meng et al., 2025; Wu et al., 2024; Ko et al., 2023). In parallel, graph neural networks (GNNs) have emerged as a natural model class for relational industrial data; attention-based graph architectures can further provide learned importance weights that are useful for interpretation (Veličković et al., 2018; Zhou et al., 2020; Liu et al., 2026). However, comparatively fewer studies have demonstrated physics-structured, tool-informed graph learning on real, large-scale industrial semiconductor equipment datasets across multiple tools and processes, under evaluation protocols that reflect actual deployment constraints.

In addition to physics-informed graph learning, recent progress in deep learning for tabular data has produced architectures that are more specialized for structured industrial datasets. TabM improves multilayer perceptron-based tabular learning through parameter-efficient ensembling, while TabR combines neural representation learning with nearest-neighbor-style retrieval from similar historical samples (Gorishniy et al., 2025, 2024, 2021). These methods provide useful complementary benchmarks for semiconductor PASS/FAIL prediction because they operate on flattened tabular representations rather than explicitly encoded physical graphs. Therefore, comparing LightGBM, TabM, TabR, and Physics-GAT makes it possible to distinguish performance gains arising from strong tabular nonlinear learning, local historical similarity, and physics-structured subsystem interactions.

In this work, we address this gap by proposing a physics-structured, attention-based GNN framework for semiconductor-tool PASS/FAIL prediction using real industrial manufacturing data. We consider two representative tool families with distinct physical mechanisms, a chemical mechanical polishing (CMP) tool family (EBR), and a dry plasma etching tool family (NXE), and we define a graph construction strategy that mimics each tool's physical modularity. Specifically, numerical sensor traces are summarized into standard statistical features and then grouped into physically meaningful subsystems (e.g., pressure/flow control, rotational/motion subsystems, vacuum and power subsystems, actuation-state subsystems), while contextual categorical variables describing production conditions and process steps are represented as additional graph tokens. Subsystem-level node embeddings are learned via attention pooling over the sensors within each subsystem, and a graph attention network models interactions among subsystems and contextual tokens. The resulting attention weights provide subsystem and within-subsystem importance scores that can be used to provide insights for fault analysis. To reflect deployment realism, we train the model on an 18-month window of industrial data using a chronological split in which the most recent 20% of time is reserved for testing, and we restrict our study to tools with more than 8000 observations over the full period to ensure statistical stability in training.

Beyond the proposed Physics-GAT model, this work also evaluates modern neural tabular architectures and their relationship to the LightGBM benchmark. TabM is used to test whether parameter-efficient neural ensembling over the same processed tabular features can provide stronger nonlinear prediction. TabR is used to test whether retrieval of similar historical training runs can improve predictions for rare or locally recurring fault signatures. We further introduce a context-aware gated ensemble of LightGBM, TabM, and TabR as a future-oriented framework for combining complementary model classes under different production contexts. Together, these models provide a broader comparison between tree-based learning, physics-structured graph learning, parameter-efficient neural ensembling, and retrieval-augmented tabular learning. Table 1 summarizes the models used in this work and their objectives.

To establish the experimental foundation and account for class imbalance across process tools, we evaluate our framework on a multi-tool dataset spanning five distinct EBR equipment families. Table 2 provides a comprehensive overview of the dataset statistics, detailing the total run sample sizes (N), absolute failure counts (N_{FAIL}), and empirical failure rates across each evaluated tool family.

This paper is organized as follows. Section 2 introduces the benchmark GBDT modeling pipeline, including single-machine training (Section 2.1) and multi-machine aggregated training (Section 2.2). Section 3 presents the proposed physics-structured GNN and the associated training procedures for single-machine (Section 3.1) and multi-machine (Section 3.2) settings. Section 4 introduces the TabM model for

Table 1
Summary of investigated models and objectives.

Model	Model Class/Category	Reason for Investigation
Feedforward Neural Network (FNN)	Classical Deep Learning	Used as an initial standalone baseline to evaluate standard parametric neural performance against the tree-based benchmark.
LightGBM	Gradient Boosting Decision Tree (GBDT)	Established as a strong tabular baseline to capture highly nonlinear relationships efficiently without deep architectures, operating directly on categorical variables and handling extreme class imbalance.
Physics-GAT	Physics-Structured Graph Attention Network	Proposed to move beyond flat feature vectors by representing runs as heterogeneous graphs to capture coordinated changes across physically coupled subsystems, suppress process noise, and provide diagnostic interpretability via internal attention layers.
Cross-Machine GNN Metahead	Multi-Source Representation Learning	Investigated to combine independent, parallel latent features from physically distinct process steps (EBR polishing and NXE/G602 dry-etching backbones). By freezing the localized structural embeddings and stacking them into a unified representation space, it allows a single joint decision head to exploit cross-process physical correlations (such as mechanical polishing stress paired with plasma-induced thermal shifts) to boost the final target tool accuracy.
TabM	Tabular Deep Learning (Ensemble-style)	Evaluated to determine if a modern, parameter-efficient neural ensemble operating on flat tables can provide stronger nonlinear predictions and reduce overfitting compared to standard deep architectures.
TabR	Tabular Deep Learning (Retrieval-augmented)	Investigated to evaluate if explicitly comparing a current production run with similar historical training runs can amplify local evidence and improve fault predictions for rare or locally recurring failure signatures.
Hybrid Neural-Tree Framework	Hybrid Representation Learning	Explores decoupling feature extraction from classification by freezing optimized deep tabular representations from TabM/TabR backbones and using them to train a LightGBM head from scratch, boosting performance on highly imbalanced data without label leakage.
Context-Aware Gated Ensemble	Instance-Level Mixture of Experts	Introduced to optimize fault detection within a single process domain by evaluating a run's compact context (recipe, step, duration) to compute dynamic, instance-specific model weights over LightGBM, TabM, and TabR.

Table 2
Summary of dataset statistics across evaluated EBR tool families.

Tool name	Total sample size (N)	Failure count (N_{FAIL})	Failure rate (%)
EBR_51	14,912	102	0.68%
EBR_52	17,838	184	1.03%
EBR_53	29,207	681	2.33%
EBR_54	28,654	495	1.72%
EBR_55	8359	138	1.65%
Total/Overall	**98,970**	**1,600**	**1.62%**

parameter-efficient tabular neural ensembling. Section 5 presents the TabR model for retrieval-augmented tabular prediction. Section 6 describes the context-aware gated ensemble of LightGBM, TabM, and TabR. Section 7 reports the experimental results and discussion, including benchmark-model performance (Section 7.1), physics-structured GNN performance and interpretability analysis (Section 7.2), and tabular-model performance for TabM and TabR (Section 7.4). Section 8 concludes the paper.

2. Benchmark GBDT model training

Before introducing the proposed graph neural network, a strong tabular-data benchmark was established using LightGBM, a gradient boosting decision tree (GBDT) implementation designed for efficient learning on high-dimensional structured data (Ke et al., 2017). For each production run, the model input consisted of run-level statistical summaries extracted from routine tool sensor traces, together with engineered numerical descriptors and contextual categorical variables describing the operating condition, such as process stage, process step, lot type, recipe, and product identity. During data preprocessing, runs with missing metrology labels or severe input incompleteness were removed, and invalid control or overflow values were corrected.

LightGBM was selected as the benchmark model for three main reasons. First, GBDT models can capture strongly nonlinear relationships in industrial data without requiring deep architectures. Second, the

LightGBM implementation is computationally efficient for the 300–600 feature scale considered in this work because it uses histogram-based split finding together with leaf-wise tree growth, which substantially reduce training time and memory cost (Ke et al., 2017). Third, LightGBM can operate directly on categorical variables and search category partitions during split construction, thereby avoiding the high-dimensional one-shot expansion that would otherwise be required for FNN models. This property is especially advantageous for semiconductor manufacturing data, where contextual variables often contain strong predictive information regarding recipe, product, and lot conditions.

For each eligible tool, the available 18-month dataset was sorted chronologically, and the most recent 20% of the timeline was reserved as a strictly held-out test set. To ensure statistical robustness and avoid overfitting during development, the remaining 80% model-development set was partitioned into an 80/20 train/validation split using random sampling (yielding an overall 64% train, 16% validation, and 20% test split across the entire dataset). A random split was deliberately selected over a chronological split within the internal development phase to prevent early-stopping criteria from over-fitting to localized process drift and short-term tool maintenance windows. By sampling broadly across the development timeline, the validation score acts as a global regularizer, ensuring selected model parameters maintain consistent operational reliability across shifting process regimes.

All hyperparameter tuning and model selection were performed exclusively on the model-development set so that the test data were not consulted during optimization. Because the FAIL rate was approximately 2%, model quality was evaluated primarily by the receiver operating characteristic area under the curve (ROC–AUC), which is well suited to strongly imbalanced binary classification (Gonçalves et al., 2014). In addition, a second industrially relevant criterion was reported: the maximum achievable true positive rate (TPR) under a false positive rate (FPR) constraint of 20%. This operating-point metric reflects the practical requirement that the classifier recover as many abnormal wafers as possible while maintaining an acceptable burden of erroneous classifications in production. False positives would let failed wafers be involved into further production steps and even could be finally detected only at the terminal testing phase, which wastes lots of resources.

For binary fault detection, the rare failure state is explicitly defined as the positive class (Class 1 = FAIL), while nominal operation represents the negative class (Class 0 = PASS). Under this formulation, a False Negative (FN) corresponds to a missed detection—where a defective wafer is misclassified as normal and allowed to proceed downstream, posing a direct yield risk. Conversely, a False Positive (FP) represents a false alarm—where a healthy wafer is misclassified as abnormal, triggering an unwarranted equipment alert and unnecessary inspection.

In an industrial production environment, a viable deployment target typically demands a TPR of $\geq$ 80% within this low-FPR operating window. However, the standalone baseline model results on individual tools fall short of this rigorous industrial threshold. This performance ceiling stems directly from the severe data constraints inherent to high-yield semiconductor lines: individual tool datasets (such as the EBR family) are often limited in size, exhibit extreme class imbalance with a nominal failure rate around 1%, and suffer from high operational noise levels. Under these data-scarce conditions, standard tabular models lack sufficient failure density to resolve a sharp, absolute decision boundary. Acknowledging this limitation underscores why the field must look beyond standard tree-based architectures toward specialized data engineering—such as physics-informed grouping, localized historical retrieval, and multi-machine feature aggregation—to systematically suppress noise, share sparse failure signatures, and drive absolute performance closer to practical deployment targets.

Table 3
Representative LightGBM hyperparameters explored for the benchmark classifier. The final selected values should be filled with the validation-optimal settings from the actual runs.

Hyperparameter	Candidate values	Selected value
objective	`binary`	`binary`
metric	`auc`	`auc`
learning_rate	{0.03, 0.05, 0.10}	0.05
max_estimators	{50, 100, 200}	200
num_leaves	{32, 64, 128, 256}	128
max_depth	{4, 6, 8, −1}	-1
min_gain_to_split	$\{0, 10^{-3}, 10^{-2}, 10^{-1}\}$	10^{-3}
lambda_l1	$\{0, 10^{-2}, 10^{-1}, 1\}$	0
lambda_l2	$\{0, 10^{-2}, 10^{-1}, 1, 10\}$	0.06
scale_pos_weight	$\{1, n_-/n_+\}$	**1**

Although the dataset is highly imbalanced, empirical testing demonstrated that using `scale_pos_weight` generally reduced validation performance relative to the unweighted setting, and the best-performing models were therefore obtained with `scale_pos_weight=1`. A likely explanation is that aggressive class reweighting changes the optimization target by forcing the learner to emphasize minority-class errors more strongly, which can improve minority recall but degrade the ranking of borderline samples. Since model selection in this work is based on ROC–AUC and performance in the low-FPR operating region, the unweighted objective yielded more favorable score ordering for deployment-oriented decision thresholds.

Selected values of hyperparameters were tuned using Optuna Bayesian optimization over candidate distributions grounded in standard literature baselines, yielding the optimal continuous and discrete configurations reported in Table 3.

2.1. Single-machine GBDT training

In the single-machine setting, one LightGBM model was trained independently for each tool using only data from that tool. Because tree-based models are largely insensitive to monotone rescaling of numerical features, no additional normalization was applied to the numerical variables in this benchmark setting. This preserves the native operating ranges of each machine and provides the simplest industrial baseline. The model-development portion of each tool dataset was further divided into training and validation subsets by random splitting, and the hyperparameters listed in Table 3 were tuned using validation performance only. The contextual variables were supplied directly to LightGBM as categorical features rather than expanded by one-hot encoding. The validation-best model configuration with early stop of max 50 estimators was then evaluated on the held-out temporal test set. This single-machine setting establishes the reference performance that can be achieved without borrowing information from other tools.

2.2. Multi-machine GBDT training

In the multi-machine setting, data from the target tool were aggregated with data from helper tools belonging to the same process family. Unlike the single-machine case, the numerical features of each tool were standardized separately before aggregation, whereas the shared categorical encodings were kept unchanged. This separated-scaling strategy was not introduced because LightGBM requires normalized inputs, but because nominally similar tools can exhibit substantial domain shift in their absolute signal levels due to calibration offsets, chamber condition, maintenance history, and local operating practice. By scaling each dataset independently, the aggregation step aligns inter-tool numerical distributions while preserving the relative within-tool relationships that remain informative for classification (Ou et al., 2025).

Because LightGBM training is computationally inexpensive, helper-tool selection was performed by exhaustive search. In this context, a helper tool is defined as a nominally similar machine from the same process family whose historical production data is standardized and aggregated with the target tool's data to broaden the training distribution and share sparse failure signatures . For a target tool with $M-1$ candidate helper tools, all $2^{M-1}-1$ non-empty helper-set combinations were evaluated. Each aggregated candidate was trained only on the pre-test data and ranked by validation performance, and the best-performing combination was selected without consulting the held-out test set. The final reported multi-machine result for each target tool therefore corresponds to the test-set performance of the validation-best aggregation scheme. This exhaustive strategy avoids data leakage while fully exploiting the practical speed advantage of GBDT models over more computationally expensive deep architectures.

Raw sensor data preprocessing was conducted by the Clean Energy Smart Manufacturing Innovation Institute (CESMII), while inline ground-truth fault labels (PASS $= 0$, FAIL $= 1$) were assigned directly by Seagate Technology based on proprietary quality criteria. Custom machine learning data aggregation scripts were executed by the authors to construct the final dataset matrices. Due to corporate non-disclosure policies, proprietary sensor naming schemes and exact metrology thresholds are omitted.

3. Graph neural network training

To move beyond purely tabular learning and to better exploit the physical organization of semiconductor tools, we developed a physics-structured graph neural network (GNN) that represents each production run as a small heterogeneous graph rather than as a flat feature vector. The central modeling idea is that process failures are rarely caused by an isolated scalar deviation; rather, they more often emerge from coordinated changes across physically coupled subsystems. Accordingly, the present implementation first transforms each run into a structured sensor tensor in which each sensor is described by a fixed set of statistical summaries and data-quality indicators, including central tendency, dispersion, extrema, range, and completeness descriptors, together with a reserved coefficient-of-variation channel when such a stability feature is physically meaningful for that sensor family. Missing or invalid values are not simply discarded; instead, they are explicitly represented through a parallel missing-mask channel so that the model can distinguish between "normal low value" and "not observed" conditions. For the EBR/CMP tools, these sensor embeddings are then organized into eight physically meaningful groups: `process_params`, `pressure`, `slurry_flow`, `rotation`, `stage_head`, `dressing_water`, `step_numbers`, and `motor_current`. This graph-level organization is complemented by an auxiliary numeric branch that consumes all pre-engineered `feat_*` variables together with retained timing and duration descriptors. In the current EBR pipeline, these derived features were deliberately chosen to encode physically interpretable quantities such as stability via coefficient-of-variation features, subsystem balance measures, left–right asymmetry indicators, pressure-distribution ratios, kinematic ratios, and duration/completeness descriptors, because these quantities often reflect physically meaningful deviations more directly than raw statistics alone. Importantly, the GNN script itself does not engineer these features online; instead, the preprocessing stage produces them once, and the graph model consumes them through a dedicated auxiliary branch, thereby separating feature engineering from graph learning and making the final workflow easier to audit in an industrial setting (Kadlec et al., 2009; Jiang et al., 2020; Sun and Ge, 2021; Ou et al., 2024, 2025). These sensor embeddings are then organized into seven physically meaningful G602 dry-etch groups: `gas_flow_chemistry`, `chamber_pressure_pumping`, `plasma_rf_source_bias`, `esc_wafer_thermal_control`, `backside_he_cooling`, `endpoint_opticalemission`, and `recipe_tool_state`. The corresponding dry-etch-derived auxiliary physics features are organized as `total_flow_gas_ratio`, `pressure_flow_residence_time`, `rf_bias_power_coupling`, `thermal_chuck_backside_he_stability`, `endpoint_slope_integral`, and `gas_pressure_plasmatemperature_interactions`. Conceptually, these G602 groups will follow the same subsystem-based philosophy by organizing signals according to ion-source, RF/plasma-power, vacuum, gas-delivery, rotation, source-health, and diagnostic subsystems rather than by exposing confidential sensor identities directly.

3.1. Single-machine GNN training

In the single-machine setting, the above physics-structured representation is used to train one graph model per tool. After preprocessing, all numerical sensor statistics and auxiliary engineered features are standardized using training-data statistics only, invalid overflow-like control values are converted to missing values, and remaining missing entries are mean-imputed only for numerical stability while their absence is preserved in the mask tensor. This standardization step is necessary here, unlike in the GBDT benchmark, because the neural architecture combines many feature channels through learned linear projections and attention operations and therefore benefits from comparable numerical scales during optimization (Ou et al., 2025). The overall workflow of GNN training is specified in Fig. 1.

For each run, the model first concatenates the standardized sensor-statistics tensor and the missing-mask tensor, projects each sensor vector into a learned hidden representation, and adds a sensor-identity embedding so that sensors occupying similar statistical ranges but different physical roles remain distinguishable. For each sensor i inside a physical group g, the model concatenates its standardized statistics vector $S_i \in \mathbb{R}^{d_s}$ and its corresponding missing-mask vector $M_i \in \{0,1\}^{d_s}$ into a unified input representation:

$$s_i = [S_i\,;\, M_i] \in \mathbb{R}^{2d_s}. \tag{1}$$

This vector is projected into a d-dimensional hidden space. To ensure that sensors with similar numerical ranges but different physical roles remain distinguishable, a learned sensor-identity embedding is added to the projected vector:

$$x_i^{(0)} = \sigma\left(W_{\text{proj}} s_i + b_{\text{proj}}\right) + e_{\text{id}}(i), \tag{2}$$

where $W_{\text{proj}} \in \mathbb{R}^{d\times 2d_s}$ is a learned projection matrix, $b_{\text{proj}} \in \mathbb{R}^d$ is a bias vector, $e_{\text{id}}(i) \in \mathbb{R}^d$ is a unique coordinate-identity embedding for sensor i, and σ represents a non-linear activation function (such as a Rectified Linear Unit). The model then performs attention pooling within each physical group so that multiple related sensors are aggregated into a single group node. This step is important for two reasons: it compresses redundant sensor-level information into subsystem-level representations, and it creates a natural interpretability interface, because the within-group attention weights indicate which sensor channels inside a given subsystem were most responsible for the final decision. The model performs attention pooling over the sensor nodes within each group $\mathcal{G}_g$ before applying a learnable gating function to filter out uninformative subsystems:

$$\tilde{h}_g^{(0)} = \sum_{i\in\mathcal{G}_g} \alpha_i x_i^{(0)} \quad \text{where} \quad \alpha_i = \text{softmax}\left(\mathbf{a}_{\text{pool}}^{\top} \tanh\left(W_{\text{pool}} x_i^{(0)} + b_{\text{pool}}\right)\right), \tag{3}$$

$$h_g^{(0)} = \sigma_{\text{sigmoid}}\left(\mathbf{w}_g^{\top} \tilde{h}_g^{(0)} + b_g\right) \odot \tilde{h}_g^{(0)}, \tag{4}$$

where $W_{\text{pool}} \in \mathbb{R}^{d\times d}$, $\mathbf{a}_{\text{pool}} \in \mathbb{R}^d$, $\mathbf{w}_g \in \mathbb{R}^d$, and $b_g \in \mathbb{R}$ are learned parameters, and $\odot$ represents the element-wise product. The attention weights α_i provide immediate diagnostic insights by finding which sensors dominate subsystem behavior. A learnable group gate is

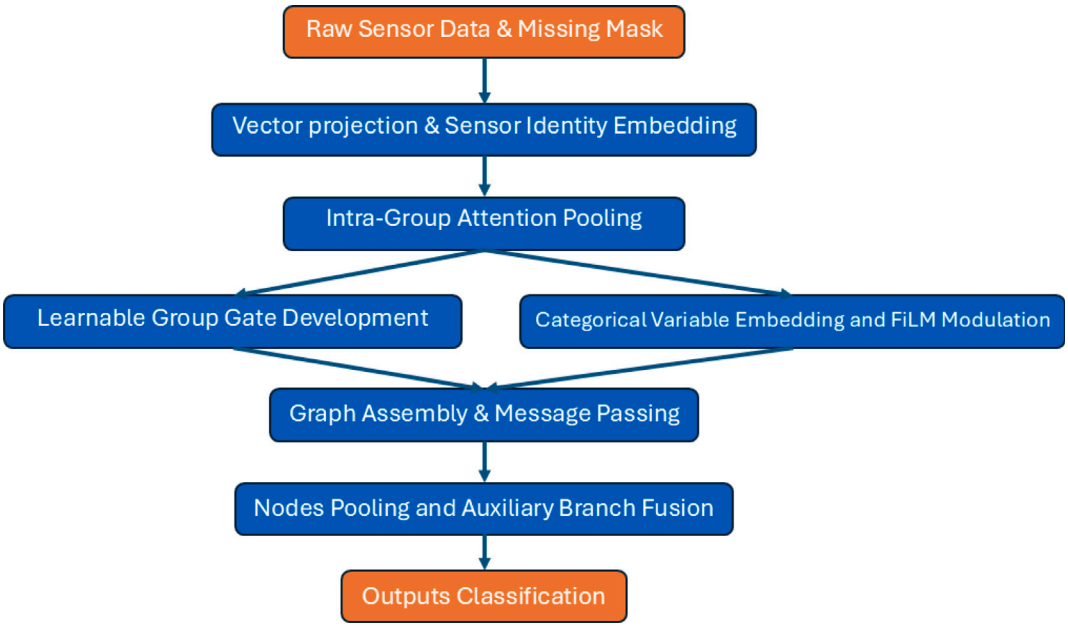

Fig. 1. Overall workflow of GNN model training.

then applied to each subsystem node, allowing the network to softly suppress groups that carry limited predictive value for a particular tool. In parallel, categorical contextual variables such as `stage`, `step`, `lottype`, `recipe`, and `prod` are embedded as token nodes. Rather than treating these contextual variables as passive side information appended at the end of the network, the model uses them twice: first, they form explicit graph nodes, and second, their joint embedding is passed through a FiLM-style conditioning block that modulates the subsystem nodes before graph propagation:

$$\hat{h}_g^{(0)} = \left(W_{\gamma,g}c + b_{\gamma,g}\right) \odot h_g^{(0)} + \left(W_{\beta,g}c + b_{\beta,g}\right), \tag{5}$$

where $W_{\gamma,g}, W_{\beta,g} \in \mathbb{R}^{d\times d_c}$ and $b_{\gamma,g}, b_{\beta,g} \in \mathbb{R}^d$ are learned affine projections. This design was chosen because the same physical subsystem can have very different "normal" operating signatures under different recipes, lot classes, or process stages, and therefore context must be allowed to re-parameterize the physical representation rather than merely shift the final classifier (Battaglia et al., 2018; Gilmer et al., 2017). The resulting graph contains subsystem nodes and categorical token nodes, connected through a dense but typed adjacency pattern with four edge classes: self edges, subsystem–subsystem edges, subsystem–token edges, and token–token edges. Because the number of graph nodes is small and because the true cross-subsystem interaction structure is only partially known a priori, a dense graph with learned edge-type bias was preferred over a hand-crafted sparse topology. Two graph-attention layers are then applied so that each subsystem representation can update itself by selectively attending to the other subsystems and to the contextual tokens (Veličković et al., 2018; Wu et al., 2021), as follows:

$$\hat{h}_u^{(l+1)} = \sigma_{\text{elu}}\left(\sum_{v\in\mathcal{N}(u)} \alpha_{uv}^{(l)} W^{(l)} \hat{h}_v^{(l)}\right), \tag{6}$$

where $W^{(l)} \in \mathbb{R}^{d\times d}$ is a shared weight matrix, $\mathbf{a}^{(l)} \in \mathbb{R}^{2d}$ is the attention vector, and $b^{(l)}_{\tau(u,v)}$ is a learned scalar bias corresponding to the edge class $\tau(u,v)$ between node u and node v. After message passing, the subsystem nodes and token nodes are separately pooled by attention, concatenated into a latent vector, and fused with the auxiliary numeric branch before final classification by a multilayer perceptron. The current EBR implementation uses a weighted binary cross-entropy objective together with an auxiliary reconstruction loss on selected statistics from the mechanically informative motor-current subsystem, i.e., $\mathcal{L} = \mathcal{L}_{\text{sup}} + \lambda_{\text{rec}}\mathcal{L}_{\text{rec}}$, so that the latent embedding is encouraged not only to discriminate PASS/FAIL outcomes but also to preserve physically meaningful structure. This auxiliary task improves regularization and, from an interpretability perspective, creates an additional diagnostic handle: large reconstruction discrepancies can signal that the latent representation has identified an unusual mechanical pattern even before thresholded classification is applied. Training is performed with mini-batches, AdamW optimization, gradient clipping, and early stopping based on validation ROC–AUC, while final assessment follows the same imbalanced-data criteria used for the benchmark model. Overall, the single-machine architecture leaves several distinct spaces for fault analysis: sensor-level attention within each group, subsystem-level group gating, cross-subsystem interaction through graph attention, contextual modulation through categorical tokens, and latent- or reconstruction-based screening of anomalous runs.

3.2. Multi-machine GNN training

The multi-machine GNN extends the same physics-structured architecture to jointly train on several EBR datasets with identical schema, while explicitly accounting for the fact that nominally similar tools often exhibit tool-to-tool shifts in absolute signal level, missingness pattern, and operating range. In the current EBR implementation, multiple parquet files belonging to the same tool family are streamed and stacked into a single aggregated training pool; the non-test portion of the combined timeline is first partitioned into train and validation subsets, and the most recent time period is kept strictly for testing, after which performance is reported both on the pooled test set and separately for each contributing tool dataset. A key design choice in this multi-machine setting is the scaling strategy. Rather than forcing a single global normalization across all machines, the default implementation uses per-dataset scaling before pooling, so that each tool contributes features expressed relative to its own historical operating distribution. This preserves within-tool physical relationships while reducing the effect of calibration offsets, maintenance-state differences, or local control biases that would otherwise blur the shared structure. At the same time, categorical vocabularies are built consistently across the pooled training data, and the graph topology itself is shared across all tools because the physical groups correspond to common subsystems

of the tool family rather than to machine-specific identifiers. This is precisely where the physics-structured formulation is especially valuable: if one were to aggregate raw sensor columns naively, domain shift could dominate the learning process, whereas grouping sensors into subsystem nodes encourages the model to learn transferable signatures such as pressure instability, flow imbalance, rotation mismatch, step-sequence anomalies, or motor-load irregularity at the subsystem level. In practice, the same graph pipeline used for single-machine learning is therefore preserved, but the training data are delivered through a streaming multi-file dataset loader so that large industrial datasets can be processed without materializing the full pooled table in memory. This makes the approach scalable enough for realistic factory data volumes while preserving strict train/validation/test separation. The multi-machine setting also enhances interpretability in a way that a flat pooled classifier cannot. Because the model yields both pooled and per-tool test metrics, and because attention values are available at the sensor, subsystem, and contextual levels, one can compare which physical groups dominate failures consistently across machines and which groups become important only for specific tools. This distinction is important for practical fault detection: common patterns may indicate family-level process mechanisms, whereas tool-specific deviations may point to maintenance issues, alignment problems, local chamber health, or calibration drift. As currently coded, this shared-model framework is implemented only for the EBR/CMP family. To demonstrate the generality of this physics-structured GNN framework, the architecture is extended to the NXE and G602 dry-etch tool families. Dry-etch processing involves complex chemical, electrical, and physical interactions that differ significantly from wet-polishing processes, requiring signals to be grouped into distinct physical subsystems to prevent domain shift across different machines in a fleet.

3.3. Cross-machine GNN metahead training

The cross-tool GNN training framework extends this methodology by enabling joint transfer learning between physically distinct process steps, specifically combining the polishing signatures of the EBR family with the dry-etching profiles of the NXE or G602 families. This multi-stage training strategy is designed to construct a shared representational space across disparate tool types that do not share a common sensor schema. The workflow operates by first training the physics-structured GNN backbones independently on each tool family to learn localized, subsystem-level interaction dynamics. Once backbone training is complete, the optimization is halted, and we extract the high-dimensional latent representation vectors from the graph readout layer, capturing the compressed physical state of the run just before it enters the tool-specific classification head. After extracting these feature vectors from both the EBR backbone and the corresponding dry-etch (either NXE or G602) backbone, we stack the resulting representation vectors into a unified feature space. This stacked feature vector is then used to train a joint, aggregated decision head that produces final classification and yield predictions for the EBR process. By evaluating this stacked framework on both EBR with G602 and EBR with NXE, the model can exploit latent physical correlations—such as matching pressure-induced mechanical stress in polishing with plasma-induced thermal shifts in etching—thereby significantly enhancing predictive accuracy and diagnostic robustness on the target EBR tools.

To clarify the operational distinction from the multi-tool sample pooling described in Section 3.2, "stacking" in this cross-machine setting refers specifically to column-wise vector concatenation of the high-dimensional latent embeddings extracted from each independently trained GAT backbone, rather than row-wise sample pooling of raw records. The joint decision head is trained in a supervised manner using the binary PASS/FAIL labels corresponding to the target EBR process runs. During inference and operational testing, evaluation is performed strictly per-tool on the EBR test dataset using only its run-specific sensor window. Because the stacked decision head requires no concurrent or downstream measurements from other equipment during runtime, this deployment setup completely avoids any risk of cross-tool or temporal data leakage.

4. TabM model training

In addition to the physics-structured graph neural network, TabM (Gorishniy et al., 2025, 2021) is evaluated as a modern deep-learning model for tabular PASS/FAIL prediction. Unlike the Physics-GAT, which explicitly encodes subsystem-level physical structure, TabM operates on a flattened tabular representation and focuses on improving the performance of multilayer perceptrons (MLPs) through parameter-efficient ensembling. This makes TabM a useful complementary benchmark to assess whether strong nonlinear tabular learning alone can capture the relationships present in semiconductor manufacturing data.

TabM is based on the observation that MLP performance on tabular data can be significantly improved by ensembling. Instead of training K independent neural networks, TabM shares a common backbone while introducing lightweight member-specific factors, effectively approximating an ensemble within a single model. For each input sample i, the model produces K logits:

$$z_{i,1}, z_{i,2}, \dots, z_{i,K}, \tag{7}$$

and the final predicted probability is obtained by averaging:

$$\hat{p}_i = \frac{1}{K} \sum_{k=1}^{K} \sigma(z_{i,k}), \tag{8}$$

where $\sigma(\cdot)$ is the sigmoid function.

The TabM architecture can be interpreted as a BatchEnsemble-style model (Wen et al., 2020), where each ensemble member shares a global weight matrix while using member-specific multiplicative factors. For a hidden layer ℓ, the transformation can be written as:

$$h_{i,k}^{(\ell+1)} = \phi\left(\left(h_{i,k}^{(\ell)} \odot r_k^{(\ell)}\right) W^{(\ell)} \odot s_k^{(\ell)} + b_k^{(\ell)}\right), \tag{9}$$

where $r_k^{(\ell)}$ and $s_k^{(\ell)}$ are member-specific parameters, and $\phi(\cdot)$ is a nonlinear activation function.

For each production run, the TabM input is constructed using the same processed features as the neural baselines. Sensor statistics are arranged into a tensor

$$A_i \in \mathbb{R}^{S \times Q}, \tag{10}$$

where S is the number of sensors and Q is the number of statistical descriptors. A missing-mask tensor

$$M_i \in \{0,1\}^{S \times Q} \tag{11}$$

is constructed in parallel. After standardization and imputation using training-only statistics, the numerical input is defined as

$$x_i^{(\text{num})} = \left[\text{vec}(A_i),\ \text{vec}(M_i),\ u_i\right], \tag{12}$$

where u_i includes all engineered `feat_*` variables and auxiliary descriptors such as processing time and duration.

Categorical variables are embedded as:

$$x_i^{(\text{cat})} = (\texttt{stage}, \texttt{step}, \texttt{lottype}, \texttt{recipe}, \texttt{prod}), \tag{13}$$

and mapped to embedding vectors that are concatenated with the numerical input.

The model is trained using a weighted binary cross-entropy loss:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \frac{1}{K} \sum_{k=1}^{K} \text{BCEWithLogits}(z_{i,k}, y_i; w_+), \tag{14}$$

where w_+ compensates for the low FAIL rate.

The same temporal split as in the benchmark models is used. The most recent fiscalweek segment is held out as the test set, while the

earlier portion is split into training and validation subsets. All preprocessing steps, including scaling and categorical encoding, are performed using training data only. Model selection is based on validation ROC–AUC, and final evaluation includes both ROC–AUC and the maximum achievable TPR under the constraint FPR ≤ 0.2.

TabM provides a strong neural baseline with several advantages. First, it can effectively model nonlinear relationships in high-dimensional tabular data. Second, its parameter-efficient ensemble structure reduces overfitting compared with single MLP models. Third, it does not rely on predefined graph structure, making it complementary to Physics-GAT. Because its inductive bias differs from both LightGBM and graph-based models, TabM is particularly valuable as a component in hybrid ensemble and meta-learning frameworks.

However, TabM does not explicitly encode physical relationships between subsystems, which limits its interpretability compared with Physics-GAT. Therefore, it is used in this work as a benchmark and as a complementary model rather than a replacement for physics-informed architectures.

5. TabR model training

To further evaluate whether modern neural tabular architectures can improve semiconductor PASS/FAIL prediction beyond conventional gradient-boosted decision trees and physics-structured graph learning, we also consider TabR, a retrieval-augmented tabular deep-learning model. Unlike TabM, which improves MLP performance through parameter-efficient ensembling, TabR introduces a local retrieval mechanism into a neural network. For a target production run, the model retrieves similar historical training runs and uses their representations, and optionally their labels, to improve the prediction for the target run. This design is motivated by the observation that tabular prediction problems often contain local structure: samples that are close in feature space may share related operating regimes, process conditions, or failure mechanisms. TabR was proposed as a simple retrieval-based tabular deep-learning model and was demonstrated to achieve strong performance on public tabular benchmarks, including datasets where gradient-boosted decision trees remain highly competitive (Gorishniy et al., 2024, 2021; Ke et al., 2017).

In the semiconductor manufacturing setting considered here, the retrieval principle is especially relevant because certain PASS/FAIL signatures may recur under similar tool, recipe, step, or sensor-state conditions. A global model such as LightGBM or an MLP must learn such patterns only through its fitted parameters, whereas a retrieval-augmented model can explicitly compare a new run with previous runs that exhibit similar feature signatures. This makes TabR a useful complementary model to the Physics-GAT: the graph model emphasizes physically structured subsystem interactions, while TabR emphasizes local historical similarity in the processed tabular feature space.

5.1. Input representation and retrieval bank

The TabR input is constructed using the same processed representation used for the other neural tabular models. For production run i, the processed sensor statistics are arranged into a fixed-width tensor

$$A_i \in \mathbb{R}^{S\times Q}, \tag{15}$$

where S denotes the number of discovered sensor IDs and Q denotes the number of statistical channels. These channels include central tendency, dispersion, extrema, range, valid-length descriptors, and coefficient-of-variation features when applicable. A corresponding missing-mask tensor

$$M_i \in \{0,1\}^{S\times Q} \tag{16}$$

is constructed in parallel, where a value of one indicates that the corresponding sensor statistic is missing or invalid.

As in the Physics-GAT and TabM pipelines, invalid overflow-like control values are first converted to missing values. Numerical statistics are then standardized using training-set means and standard deviations only. Missing entries are imputed by the corresponding training-set mean for numerical stability, while the level of missing information is retained through M_i. The final numerical input is defined as

$$x_i^{(\mathrm{num})} = \left[\mathrm{vec}(A_i),\ \mathrm{vec}(M_i),\ u_i\right], \tag{17}$$

where u_i contains all engineered `feat_*` variables and retained auxiliary descriptors such as

`processed_time`, `len`, and `duration_sec` when present.

Categorical variables are represented as

$$x_i^{(\mathrm{cat})} = (\texttt{stage}, \texttt{step}, \texttt{lottype}, \texttt{recipe}, \texttt{prod}), \tag{18}$$

with fallback column names used when the canonical names are not present. Each categorical variable is mapped to a learned embedding, and the embedded categorical representation is concatenated with the numerical input before being passed through the neural encoder.

A key component of TabR is the train-only retrieval bank. Let

$$\mathcal{B} = \{(x_j^{(\mathrm{num})}, x_j^{(\mathrm{cat})}, y_j) : j \in \mathcal{I}_{\mathrm{train}}\} \tag{19}$$

denote the set of available candidate runs in the training set. Only training samples are inserted into this bank. Validation and test samples are allowed to retrieve neighbors from the training bank, but their own labels and other validation/test labels are never inserted into the retrieval bank. This constraint is important for preventing label leakage under the temporal test protocol.

5.2. Retrieval-augmented prediction

The model first maps the target run into a latent representation using a feed-forward neural encoder:

$$h_i = f_\theta\left(x_i^{(\mathrm{num})}, x_i^{(\mathrm{cat})}\right). \tag{20}$$

For retrieval, each sample is also mapped to a retrieval feature vector ρ_i. In the implementation, ρ_i is constructed from the standardized numerical features and encoded categorical context, followed by normalization. For a target run i, the model retrieves the m most similar training-bank samples according to a similarity score $s(\rho_i, \rho_j)$:

$$\mathcal{N}_m(i) = \mathrm{TopK}_{j\in\mathcal{I}_{\mathrm{train}}}\, s(\rho_i, \rho_j), \tag{21}$$

where m is the retrieval context size. In the present implementation, cosine similarity is used in the fixed retrieval space:

$$s(\rho_i, \rho_j) = \frac{\rho_i^\top \rho_j}{\|\rho_i\|_2 \|\rho_j\|_2}. \tag{22}$$

When the queried sample is itself a training sample, its own index is excluded from the retrieved neighbor set so that the model cannot trivially recover its label from the retrieval bank.

After retrieving the neighbor set $\mathcal{N}_m(i)$, TabR uses an attention-like aggregation mechanism to extract useful information from the retrieved samples. This follows the broader idea that attention can be used to weight relevant context objects according to learned similarity (Vaswani et al., 2017). For each retrieved neighbor $j \in \mathcal{N}_m(i)$, its hidden representation is

$$h_j = f_\theta\left(x_j^{(\mathrm{num})}, x_j^{(\mathrm{cat})}\right). \tag{23}$$

The target representation produces a query vector, and each neighbor produces a key vector:

$$q_i = W_q h_i, \qquad k_j = W_k h_j. \tag{24}$$

The attention weight assigned to neighbor j is then

$$\alpha_{ij} = \frac{\exp\left(q_i^\top k_j / \sqrt{d_k}\right)}{\sum_{\ell\in\mathcal{N}_m(i)} \exp\left(q_i^\top k_\ell / \sqrt{d_k}\right)}, \tag{25}$$

where d_k is the key dimension.

Because the retrieved objects are labeled training examples, the neighbor label can also be used as part of the retrieved value representation. In the implementation, the label is projected into the hidden space and combined with the neighbor representation before value aggregation:

$$v_j = W_v \left(h_j + W_y y_j \right). \tag{26}$$

The retrieved context vector is then computed as

$$c_i = \sum_{j \in \mathcal{N}_m(i)} \alpha_{ij} v_j. \tag{27}$$

Finally, the target representation and retrieved context are combined using both direct and interaction features:

$$g_i = \left[h_i,\ c_i,\ h_i \odot c_i,\ |h_i - c_i| \right], \tag{28}$$

where $\odot$ denotes elementwise multiplication. The final logit is obtained from a classifier head:

$$z_i = \psi_\omega(g_i), \tag{29}$$

and the predicted FAIL probability is

$$\hat{p}_i = \sigma(z_i). \tag{30}$$

This retrieval-augmented structure can be interpreted as combining the representation-learning ability of a neural network with the local adaptivity of nearest-neighbor methods (Cover and Hart, 1967; Gorishniy et al., 2024). If a target run resembles previous abnormal or borderline runs, the retrieved context can help amplify this local evidence even when the global classifier is uncertain.

5.3. Training and evaluation protocol

The TabR model is trained using the same data separation protocol as the other models in this work. For each EBR tool, the most recent fiscalweek-defined portion of the data is reserved as the held-out test set, and the earlier model-development portion is split into training and validation subsets. Numerical scaling, missing-value imputation statistics, categorical vocabularies, and the retrieval bank are all constructed from training data only. The validation set is used for model selection and early stopping, while the test set is used only for final evaluation.

The training objective is weighted binary cross-entropy:

$$\mathcal{L}_{\text{TabR}} = \frac{1}{N} \sum_{i=1}^{N} \text{BCEWithLogits}(z_i, y_i; w_+), \tag{31}$$

where w_+ is the positive-class weight used to compensate for the low FAIL rate. The model is trained with mini-batch optimization, AdamW updates, dropout, gradient clipping, and early stopping based on validation ROC–AUC. In the current implementation, the retrieval context size is set to $m = 32$, so each target run attends to at most 32 retrieved training examples.

TabR has several potential advantages for the semiconductor PASS/FAIL task. First, it can exploit local recurrence in failure signatures: if rare failures occupy small pockets of the sensor-context feature space, retrieval can directly expose the model to similar historical cases. Second, unlike purely parametric tabular models, TabR can adapt predictions based on the neighborhood of each individual run. Third, because the retrieved neighbors come from the training set, the model provides a natural diagnostic interface: engineers can inspect which historical runs were retrieved for a high-risk prediction. This property is especially useful in industrial fault detection, where identifying similar previous events may support practical root-cause analysis.

At the same time, TabR is sensitive to the quality of the retrieval space. If tool drift, recipe changes, or maintenance events cause older runs to become poor analogs for current runs, retrieval can introduce misleading local evidence. For this reason, TabR is evaluated under the same strict temporal test protocol used for LightGBM, TabM, and Physics-GAT. In the present study, TabR is therefore treated as a complementary neural benchmark and as a candidate expert for future hybrid meta-learning rather than as a replacement for physics-structured graph modeling. Its performance is assessed using ROC–AUC and the maximum achievable TPR under the industrial constraint FPR ≤ 0.2, consistent with the other models in this work.

The core motivation behind decoupling representation learning from tree-based classification is to leverage the specialized operational efficiencies of both paradigms. Deep neural architectures like TabM and TabR are explicitly designed to tackle complex, high-dimensional tabular data and map raw features into rich, non-linear latent representations. Conversely, gradient-boosted decision trees such as LightGBM excel at determining optimal, non-linear decision boundaries for classification tasks, especially when dealing with highly imbalanced semiconductor process data. Decoupling these processes and allowing each component to focus entirely on its primary capability—letting the neural backbones handle representation learning while delegating the classification task to the gradient-boosted tree head—not only avoids the high computational overhead of joint end-to-end training but also significantly boosts overall predictive performance and generalization.

5.4. Hybrid tabular neural-tree framework: TabM/TabR and LightGBM integration

To combine the strong representational capabilities of modern deep tabular models with the highly efficient decision boundaries of gradient-boosted decision trees, we introduce a hybrid neural-tree framework that integrates TabM and TabR with LightGBM. This hybrid approach begins by training the feature-extraction weights of both the TabM and TabR backbones independently on the processed tabular sensor datasets. Once these backbone weights are fully optimized, we freeze them, halting further training before the data reaches their respective classification or retrieval decision heads. We then use the frozen backbones to extract static, high-dimensional latent representation vectors for each production run. Finally, we use those extracted feature vectors to train a LightGBM model entirely from scratch. By training the LightGBM decision head from the very beginning on these frozen neural representations, we successfully prevent downstream data leakage during the tree-fitting process while exploiting the robust splitting capabilities of gradient boosting on highly imbalanced semiconductor data.

6. Context-aware gated ensemble of LightGBM, TabM, and TabR

The single-model results demonstrate that different tabular learners can be favorable under different tool and operating conditions. LightGBM provides a strong tree-based benchmark, TabM provides a parameter-efficient neural ensemble, and TabR provides a retrieval-augmented local-similarity model. Because these three models have different inductive biases, a simple unweighted average of their predicted probabilities may not fully exploit their complementarity. We therefore introduce a context-aware gated ensemble that learns sample-dependent weights over LightGBM, TabM, and TabR. This approach is related to stacked generalization and super learner methods, where a second-level model combines the outputs of multiple base learners (Wolpert, 1992; Van der Laan et al., 2007). It is also closely related to mixture-of-experts models, in which a gating network assigns different weights to different experts depending on the input context (Jacobs et al., 1991).

Let the set of base models be

$$\mathcal{M} = \{\text{LGBM}, \text{TabM}, \text{TabR}\}. \tag{32}$$

Each base model is trained independently using the same training/validation/test split as described in the previous sections. LightGBM represents the classical GBDT benchmark (Ke et al., 2017), TabM represents the parameter-efficient neural ensemble model (Gorishniy et al.,

2025), and TabR represents the retrieval-augmented tabular neural model (Gorishniy et al., 2024). After training, each model produces a predicted FAIL probability for run i:

$$p_i^{(m)} = P_m(y_i = 1 \mid x_i), \qquad m \in \mathcal{M}. \tag{33}$$

The predicted probabilities are first clipped for numerical stability and converted to logits:

$$\ell_i^{(m)} = \log\left(\frac{\tilde{p}_i^{(m)}}{1-\tilde{p}_i^{(m)}}\right), \qquad \tilde{p}_i^{(m)} = \mathrm{clip}\left(p_i^{(m)}, \epsilon, 1-\epsilon\right), \tag{34}$$

where ϵ is a small constant.

6.1. Context features for the gating network

The gating network is designed to determine which base model should be trusted more for a given production run. Rather than using the full high-dimensional sensor feature vector, the gate uses a compact set of contextual features to reduce overfitting. The categorical context includes

$$c_i^{(\mathrm{cat})} = (\texttt{stage}, \texttt{step}, \texttt{lottype}, \texttt{recipe}, \texttt{prod}), \tag{35}$$

with fallback column names used when the canonical names are absent. In the multi-tool setting, a dataset or tool identifier can also be included as an additional categorical feature. The numerical context includes retained timing and duration descriptors such as `processed_time`, `len`, and `duration_sec`, together with a sensor-missingness fraction when available:

$$c_i^{(\mathrm{num})} = [\texttt{processed\_time}, \texttt{len}, \texttt{duration\_sec}, \texttt{sensor\_missing\_frac}]. \tag{36}$$

The categorical variables are mapped to learned embeddings, and the numerical context variables are standardized using meta-training statistics only. The resulting context representation is denoted by

$$q_i = \left[\mathrm{Emb}(c_i^{(\mathrm{cat})}), c_i^{(\mathrm{num})}\right]. \tag{37}$$

6.2. Soft gating formulation

The gated ensemble assigns a nonnegative weight to each base model for every sample. A small neural gating function $g_\eta(\cdot)$ maps the context representation q_i to a vector of model-specific scores:

$$a_i = g_\eta(q_i) \in \mathbb{R}^{|\mathcal{M}|}. \tag{38}$$

The final model weights are obtained using a softmax function:

$$w_i^{(m)} = \frac{\exp(a_i^{(m)})}{\sum_{r\in\mathcal{M}} \exp(a_i^{(r)})}, \qquad \sum_{m\in\mathcal{M}} w_i^{(m)} = 1. \tag{39}$$

Thus, the gate can learn, for example, to place more weight on LightGBM for one recipe regime, more weight on TabR for samples that benefit from local historical similarity, and more weight on TabM for regimes where the neural tabular representation is more reliable. Unlike a hard model-selection rule, the softmax gate allows multiple models to contribute simultaneously while still permitting an uninformative model to receive a near-zero weight.

Before combining the base logits, the meta model applies a learnable affine calibration to each base-model logit:

$$\tilde{\ell}_i^{(m)} = \alpha_m \ell_i^{(m)} + \beta_m, \tag{40}$$

where α_m and β_m are model-specific scale and bias parameters. This calibration is useful because different base models may produce probabilities with different levels of sharpness or bias. The final ensemble logit is then

$$z_i^{(\mathrm{ens})} = b_0 + \sum_{m\in\mathcal{M}} w_i^{(m)} \tilde{\ell}_i^{(m)}, \tag{41}$$

where b_0 is a global bias term. The final ensemble predicted probability is

$$\hat{p}_i^{(\mathrm{ens})} = \sigma\left(z_i^{(\mathrm{ens})}\right). \tag{42}$$

For comparison, a simple average ensemble would instead use

$$\hat{p}_i^{(\mathrm{avg})} = \frac{1}{|\mathcal{M}|} \sum_{m\in\mathcal{M}} p_i^{(m)}, \tag{43}$$

which assigns the same model weights to every sample. The gated formulation is more flexible because the weights depend on the operating context.

6.3. Meta-learner training protocol

The base models are trained first and then frozen. Their validation and test predictions are saved in row-aligned form, with identical target labels and identical validation/test row ordering across LightGBM, TabM, and TabR. The meta learner is trained only after these base predictions are generated. In the present implementation, the validation predictions from the base models serve as the meta-training set, and the held-out temporal test predictions serve as the final meta-test set. This holdout stacking protocol avoids using test labels during meta-model training and is consistent with the strict held-out evaluation design used throughout the paper.

For meta-training, the supervised loss is weighted binary cross-entropy:

$$\mathcal{L}_{\mathrm{ens}} = \frac{1}{N_{\mathrm{meta}}} \sum_{i=1}^{N_{\mathrm{meta}}} \left[-w_+ y_i \log\left(\hat{p}_i^{(\mathrm{ens})}\right) - (1-y_i)\log\left(1-\hat{p}_i^{(\mathrm{ens})}\right)\right], \tag{44}$$

where w_+ compensates for the low FAIL rate. The meta learner is optimized with AdamW, dropout, gradient clipping, and early stopping based on an internal validation split of the meta-training set. After the best epoch is selected, the final meta learner can be retrained on the full validation-prediction set for the selected number of epochs before evaluation on the held-out temporal test set.

This procedure differs from selecting the single best model by validation performance. Instead of choosing one model globally, the gated ensemble learns a context-dependent combination rule. This is important for semiconductor manufacturing data because tool behavior, recipe conditions, and process stages can influence which model family is most reliable. The learned gate weights also provide a useful diagnostic output: average weights can be reported by tool, recipe, or process condition to determine whether the ensemble is relying primarily on LightGBM, TabM, or TabR in different regimes.

6.4. Interpretation and practical advantages

The proposed gated ensemble has several practical advantages. First, it combines three complementary model classes: LightGBM captures nonlinear tree-based partitions, TabM provides a smooth neural ensemble over dense tabular features, and TabR provides local retrieval-based evidence from similar historical runs. Second, the gating network avoids the limitation of uniform averaging by allowing the model weights to vary across production contexts. Third, the affine logit calibration layer allows the meta learner to correct for differences in probability calibration across base models. Finally, the learned weights provide an interpretable summary of model reliance. If one model is consistently unhelpful, the gate can assign it a very small average weight; if a model is useful only for certain tools or recipes, the gate can preserve that local contribution rather than removing the model globally.

The final gated ensemble is evaluated using the same criteria as the individual models: ROC–AUC and the maximum achievable TPR subject to FPR ≤ 0.2. This ensures that the ensemble is assessed both as a ranking model and as a deployment-oriented fault-detection model operating under an industrial false-alarm constraint.

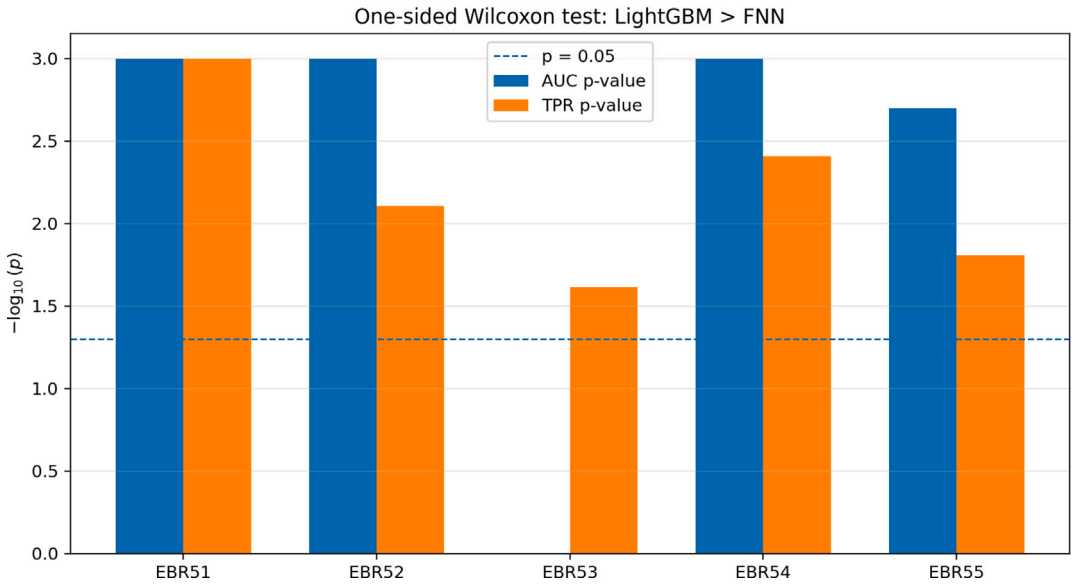

Fig. 2. One-sided Wilcoxon signed-rank test results for the hypotheses that LightGBM outperforms FNN in ROC–AUC and in TPR under FPR ≤ 0.2. Bars above the dashed line correspond to $p < 0.05$. The AUC advantage of LightGBM is significant on four of the five tools, whereas the TPR advantage is significant on all five tools.

7. Results and discussion

7.1. Benchmark model performance

The benchmark model in this work is LightGBM, and its performance was evaluated under the same temporally ordered test protocol used throughout the paper. Specifically, the most recent portion of the data (fiscalweek > 202552) was held out as a fixed test set, while the earlier portion was used for model development. However, within this model development period, the training and validation subsets were generated by random splitting. As a result, the benchmark performance is not completely deterministic, even when the temporal test set is fixed. The randomness in the train/validation split changes the exact set of samples seen during training, affects the train-only scaling parameters used in the neural baseline and in the multi-tool aggregation workflow, and also changes the validation trajectory used for early stopping. To make the comparison more statistically credible, all single-tool experiments were repeated with 10 different random seeds, and the reported values are the mean and 95% confidence interval across these runs.

To determine whether the observed benchmark differences were systematic rather than incidental, we applied a paired one-sided Wilcoxon signed-rank test (Wilcoxon, 1945) to the 10-seed results for each tool. For ROC–AUC, the alternative hypothesis was that LightGBM outperforms FNN on four of the five EBR tools (EBARA Chemical Mechanical Polishing Tool). The resulting p-values, shown in Fig. 2, strongly support LightGBM on EBR51 ($p = 0.0010$), EBR52 ($p = 0.0010$), EBR54 ($p = 0.0010$), and EBR55 ($p = 0.0020$), while EBR53 clearly favors the FNN baseline instead ($p = 1.0000$ for the one-sided test in the LightGBM-better direction). For the TPR metric under the FPR ≤ 0.2 constraint, the one-sided hypothesis that LightGBM is better than FNN is supported on all five tools, with p-values of 0.0010, 0.0078, 0.0244, 0.0039, and 0.0156 for EBR51–EBR55, respectively. Therefore, although the FNN can occasionally produce a better global ranking score on an individual tool, the LightGBM benchmark is more reliable in the practically relevant low-FPR fault-detection regime and should be regarded as the stronger baseline for the remainder of this study.

To empirically verify the validity of the internal random development split, a sensitivity diagnostic study was conducted comparing the baseline LightGBM performance under the internal random validation split against a strictly chronological validation split within the 80% development period. The empirical results demonstrate that a chronological validation split induces significant distribution shift and severe local overfitting during hyperparameter selection, yielding an extreme gap between training and validation ROC–AUC (0.9420 vs. 0.7494). In contrast, the internal random development split maintains consistent, representative feature distributions across the training and validation sets (0.9959 vs. 0.9317). By providing a stable validation signal, the random development split prevents premature or volatile early stopping on localized operational anomalies, ensuring that hyperparameter tuning optimizes for long-term generalizability on the ultimate temporal test set.

We next evaluated whether LightGBM could benefit from multi-tool aggregation. For each target tool, the best aggregation partner was selected from the validation-stage search, yielding the pairs EBR51+EBR54, EBR52+EBR55, EBR53+EBR52, EBR54+EBR52, and EBR55+EBR54. The *front* and *back* results shown in Figs. 3 and 4 correspond to the two stacking orders used during pooled training. As expected, these two orders are statistically equivalent in principle and differ only through the realized train/validation split after aggregation, so their results are close but not identical. The overall trend is nevertheless clear. Aggregation provides large and practically meaningful gains for EBR53, EBR54, and EBR55. For example, on EBR54 the mean ROC–AUC increases from 0.6000 in the single-tool setting to 0.6829–0.7068 after aggregation with EBR52, while the mean TPR increases from 0.2444 to 0.3444–0.4000. On EBR55, aggregation with EBR54 increases the mean ROC–AUC from 0.4351 to 0.5809–0.6150 and the mean TPR from 0.0476 to 0.3000–0.3619. EBR53 also improves consistently, with mean ROC–AUC rising from 0.4822 to 0.5032–0.5333 and mean TPR rising from 0.1556 to 0.1876–0.2395. By contrast, EBR52 shows only modest and statistically uncertain improvement, and EBR51 exhibits a mixed pattern: aggregation with EBR54 increases TPR but does not improve ROC–AUC relative to the single-tool benchmark. Taken together, these results demonstrated that aggregation is not universally beneficial, but it is clearly worth trying on CMP tools, especially when the target tool has limited standalone signal quality or can benefit from structurally similar helper tools, which is consistent with previous findings on industrial multi-machine aggregation (Ou et al., 2025).

The statistical significance of the aggregation benefit was again evaluated with a paired one-sided Wilcoxon signed-rank test, now using

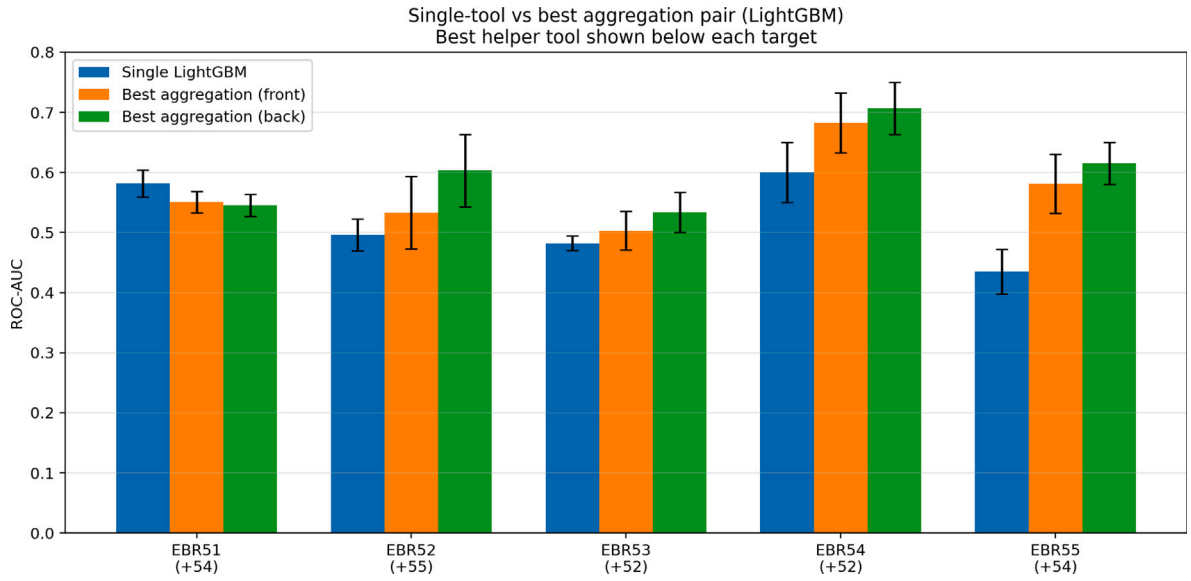

Fig. 3. Comparison between single-tool LightGBM and the best validation-selected aggregation pair for each target tool. The *front* and *back* bars correspond to the two stacking orders used during pooled training. Error bars denote the 95% confidence interval over 10 random seeds. Aggregation yields clear ROC–AUC improvements for EBR53, EBR54, and EBR55, mixed behavior for EBR51, and uncertain behavior for EBR52.

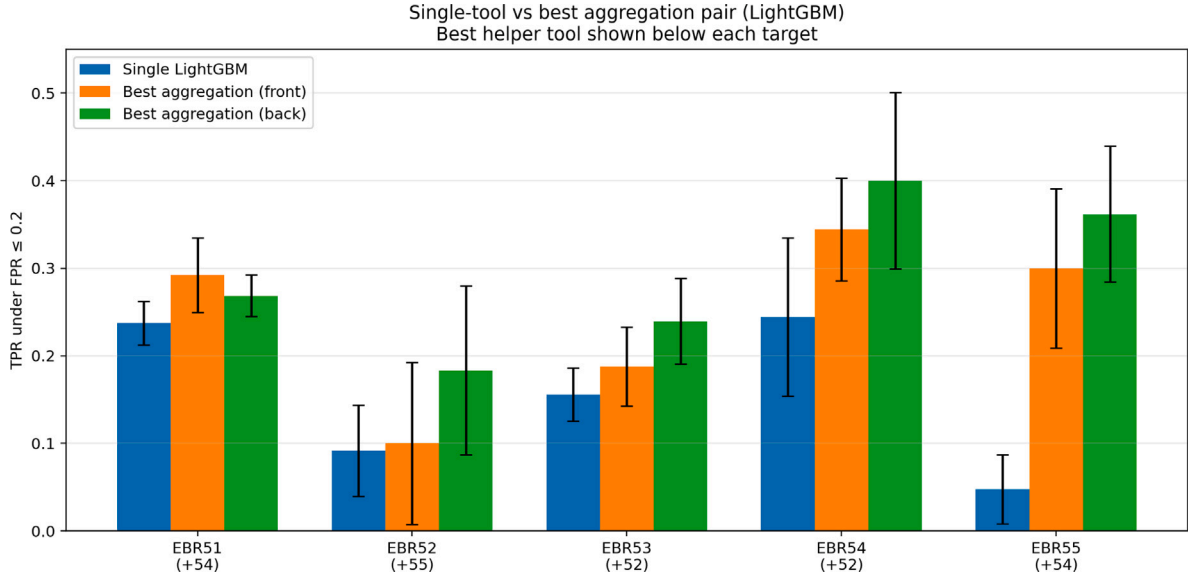

Fig. 4. Comparison between single-tool LightGBM and the best validation-selected aggregation pair for each target tool using the industrial operating metric of maximum TPR under the constraint FPR ≤ 0.2. The helper tool chosen for each target is indicated below the x-axis label. Aggregation substantially improves TPR on EBR53, EBR54, and EBR55, and also raises TPR on EBR51 and EBR52, although the latter improvements are not always statistically significant.

the alternative hypothesis that the best aggregation setting is better than the corresponding single-tool LightGBM baseline. The resulting p-values are summarized in Fig. 5. For EBR53, aggregation significantly improves both ROC–AUC ($p = 0.0029$) and TPR ($p = 0.0010$). The same conclusion holds for EBR54, with $p = 0.0010$ for ROC–AUC and $p = 0.0332$ for TPR, and for EBR55, with $p = 0.0010$ for both metrics. For EBR52, the p-values are 0.2158 for ROC–AUC and 0.4531 for TPR, indicating no statistically reliable advantage for aggregation. For EBR51, aggregation significantly improves TPR ($p = 0.0391$) but not ROC–AUC; in fact, the one-sided AUC test strongly rejects the hypothesis that aggregation is better ($p = 0.9677$), so the single-tool model remains preferable when overall ranking performance is prioritized. Therefore, the benchmark evidence supports the following practical conclusion: multi-tool LightGBM aggregation is a worthwhile strategy for CMP soft sensing and fault classification, but its benefit is dataset dependent. In the present study, aggregation is clearly beneficial for EBR53, EBR54, and EBR55, essentially neutral for EBR52, and not favorable for EBR51 in terms of ROC–AUC.

When evaluated against rigorous industrial deployment criteria, these benchmark results reveal a critical limitation. While LightGBM demonstrates a statistically significant advantage over standard feed-forward neural networks in the low-FPR operating region, its absolute True Positive Rate (TPR) falls substantially short of the ≥ 0.8 threshold typically required for live manufacturing deployment. This performance ceiling is largely imposed by the severe data constraints inherent to high-yield semiconductor lines, where extreme class imbalance (a $\sim$ 1% failure rate) and high process noise prevent standard tabular models from resolving sharp decision boundaries on a

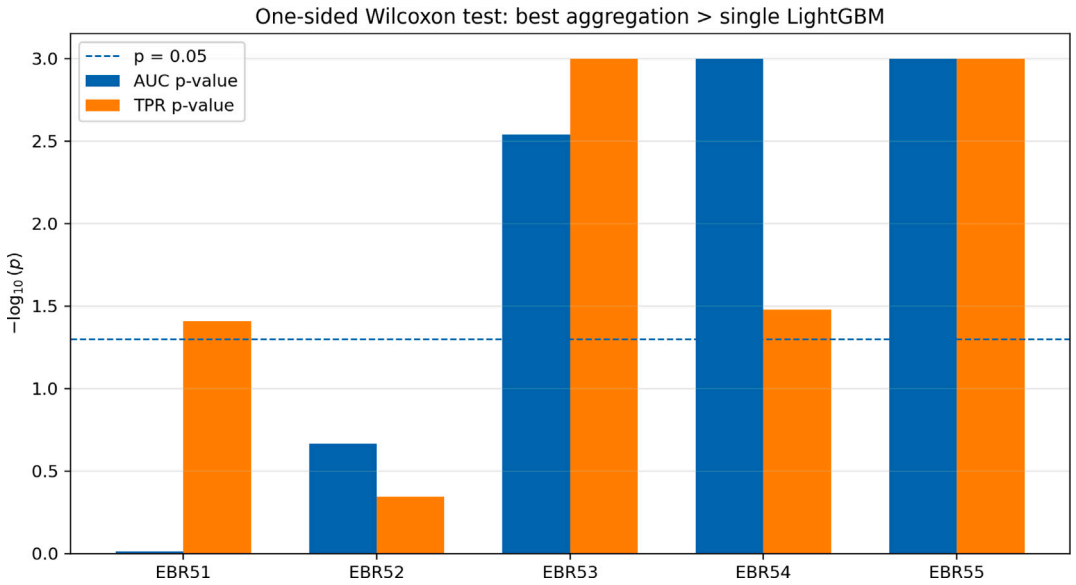

Fig. 5. One-sided Wilcoxon signed-rank test results for the hypotheses that the best aggregation setting outperforms the corresponding single-tool LightGBM baseline in ROC–AUC and in TPR under FPR ≤ 0.2. Bars above the dashed line correspond to $p < 0.05$. Aggregation is strongly supported for EBR53, EBR54, and EBR55, inconclusive for EBR52, and unfavorable for EBR51 in ROC–AUC.

single machine. While multi-tool aggregation successfully lifts this ceiling for specific machines like EBR54 and EBR55, the improvements are highly dataset-dependent and remain insufficient across the entire fleet. These baseline shortfalls highlight a stark reality that standard tree-based optimization on individual tools is not robust enough for standalone industrial use. This operational gap provides the primary motivation for the remainder of this study, driving the necessity for specialized architectures—such as physics-structured GNNs and retrieval-augmented tabular models—to systematically suppress noise, share sparse failure signatures, and move closer to practical deployment targets.

7.2. Physics-structured GNN model performance

The physics-structured graph attention network (Physics-GAT) was evaluated under the same temporal test protocol and 10-seed repeated train/validation splitting strategy used for the benchmark models. In this subsection, the reported p-values correspond to one-sided Wilcoxon signed-rank tests with the alternative hypothesis that the Physics-GAT model is better than the LightGBM benchmark. For the single-tool setting, this means *single-tool Physics-GAT* versus *single-tool LightGBM*. For the dual-tool setting, this means the *validation-selected best dual-tool Physics-GAT configuration* versus the -validation-selected best dual-tool LightGBM configuration. The same two evaluation metrics are used throughout: ROC–AUC and the maximum achievable TPR under the industrial constraint FPR ≤ 0.2.

Beyond quantitative classification performance, a primary motivation for adopting the Physics-GAT framework is diagnostic interpretability. Standard tabular baselines act as black boxes, offering little insight into feature importance during dynamic failure events. By embedding physical subsystem topologies directly into the graph attention mechanism, the model is designed to provide transparent, inspectable decision pathways. To evaluate whether this physical structure yields actionable diagnostic visibility, we examine the intra-group attention weight distributions (α_i) across operational regimes.

To evaluate the diagnostic interpretability of the Physics-GAT architecture, an empirical experiment was conducted comparing intra-group attention weights (α_i) across nominal (PASS) and abnormal (FAIL) production runs. Building on our prior observation that global subsystem gating exhibits near-uniform behavior across process groups, this analysis focused directly on the sensor-level attention distributions. As illustrated in Fig. 6, the intra-group attention weights remain virtually identical between PASS and FAIL cases. Rather than providing dynamic feature re-weighting during failure events, the network learns a near-static attention profile across operational regimes. While this uniform behavior indicates that the physics-informed structure does not yield performance enhancements on this specific dataset, it highlights the primary utility of model transparency. By opening the internal attention mechanisms to direct inspection, process engineers can observe exactly why the model fails to isolate root causes. The value of the physics-structured graph lies in its diagnostic legibility—rendering its internal decision process fully transparent—even when revealing that the learned representations are insufficient for distinct fault isolation under current process conditions.

Comparing to the single-tool Physics-GAT training, a clearer benefit of the graph formulation emerges in the dual-tool setting. Here, the graph model is trained on the target tool together with one helper tool selected from the validation-stage search. The best validation-selected helper tools for Physics-GAT are EBR51–EBR54, EBR52–EBR55, EBR53–EBR51, EBR54–EBR51, and EBR55–EBR53. The corresponding front and back results, shown in Figs. 7 and 8, differ only through the realized train/validation split after aggregation and are therefore statistically equivalent views of the same dual-training strategy. Relative to the single-tool Physics-GAT baseline, dual-tool Physics-GAT expands the attainable performance range on several tools. For example, EBR52 improves from a single-tool ROC–AUC of 0.5293 to a dual-tool range of 0.5634 in both front and back orderings, while EBR54 increases from 0.3950 to 0.5207–0.5633. EBR55 also shows a noticeable AUC increase in the best dual ordering, from 0.4153 to 0.4975. For the constrained-TPR metric, the most visible dual-training gains appear on EBR52 and EBR54, where the best dual-tool values rise to 0.3500 and 0.3333, respectively, compared with 0.3000 and 0.1778 in the single-tool setting. EBR51 also shows a modest increase in the best dual ROC–AUC result, and EBR55 shows a mild improvement in constrained TPR. EBR53 remains strong in both single and dual configurations, with dual training maintaining a comparable performance level rather than fundamentally changing the result. Thus, even before comparing with LightGBM, the raw mean results suggest that the physics-structured model often benefits from access to an additional tool.

The more important comparison is whether the dual-tool Physics-GAT can outperform the best dual-tool LightGBM benchmark. This

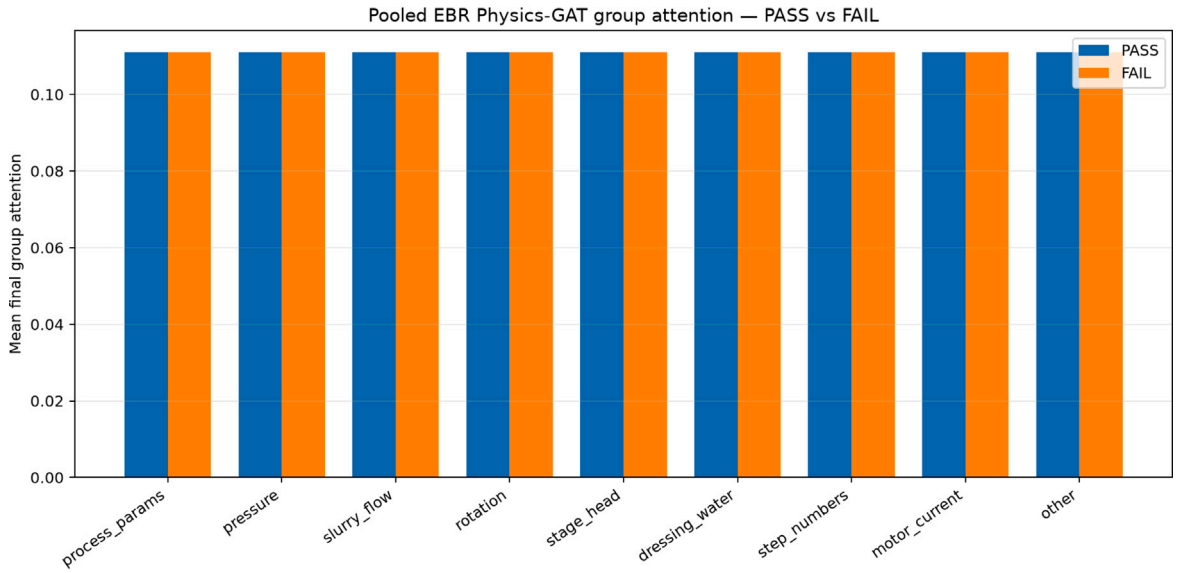

Fig. 6. Comparison of normalized intra-group attention weights (α_i) between nominal (PASS) and abnormal (FAIL) production runs across physical sensor subsystems. The static attention distributions demonstrate model transparency by revealing the architecture fails to isolate dynamic, case-specific fault signatures on this dataset.

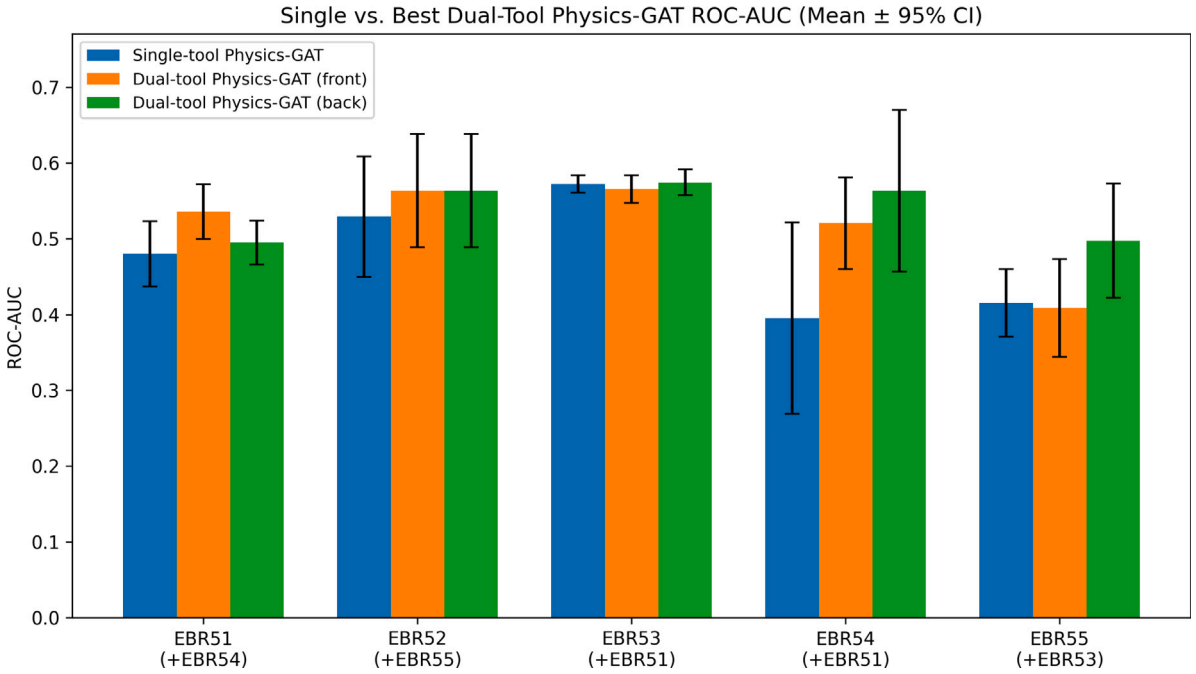

Fig. 7. Comparison between single-tool Physics-GAT and the best validation-selected dual-tool Physics-GAT configuration for each target tool. The *front* and *back* bars correspond to the two stacking orders used during pooled training. Error bars denote the reported 95% confidence intervals. Dual-tool Physics-GAT expands the attainable ROC–AUC range on several tools, especially EBR52, EBR54, and EBR55.

is evaluated in Fig. 9 using the one-sided Wilcoxon test with the alternative hypothesis that the best dual-tool Physics-GAT is better than the best dual-tool LightGBM. The resulting p-values indicate that the dual-tool graph model has a higher potential to outperform LightGBM than was observed in the single-tool setting. For ROC–AUC, the number of statistically favorable tools increases from one in the single-tool case (EBR53 only) to two in the dual-tool case, namely EBR52 (p = 0.0420) and EBR53 (p = 0.0009). For the constrained-TPR metric, dual-tool Physics-GAT is significantly better than dual-tool LightGBM on EBR52 (p = 0.0020), EBR53 (p = 0.0020), and EBR55 (p = 0.0001). In contrast, EBR51 and EBR54 do not show statistically supported dual-tool advantages for the graph model. This pattern suggests that the main strength of the physics-structured graph approach lies in its ability to exploit additional cross-tool data. Unlike LightGBM, which operates directly on flattened tabular statistics, the graph model must learn sensor-level embeddings, within-group attention, cross-group message passing, and context-conditioned subsystem interactions. This richer representation can be harder to optimize from a single tool alone, but it becomes more effective when dual-tool training provides enough data to stabilize subsystem-level relational patterns. The fact that the best helper tools selected by Physics-GAT are not always the same as those selected by LightGBM further suggests that the graph model is exploiting transferability at the level of shared physical structure rather than only marginal similarity in tabular feature distributions. Overall,

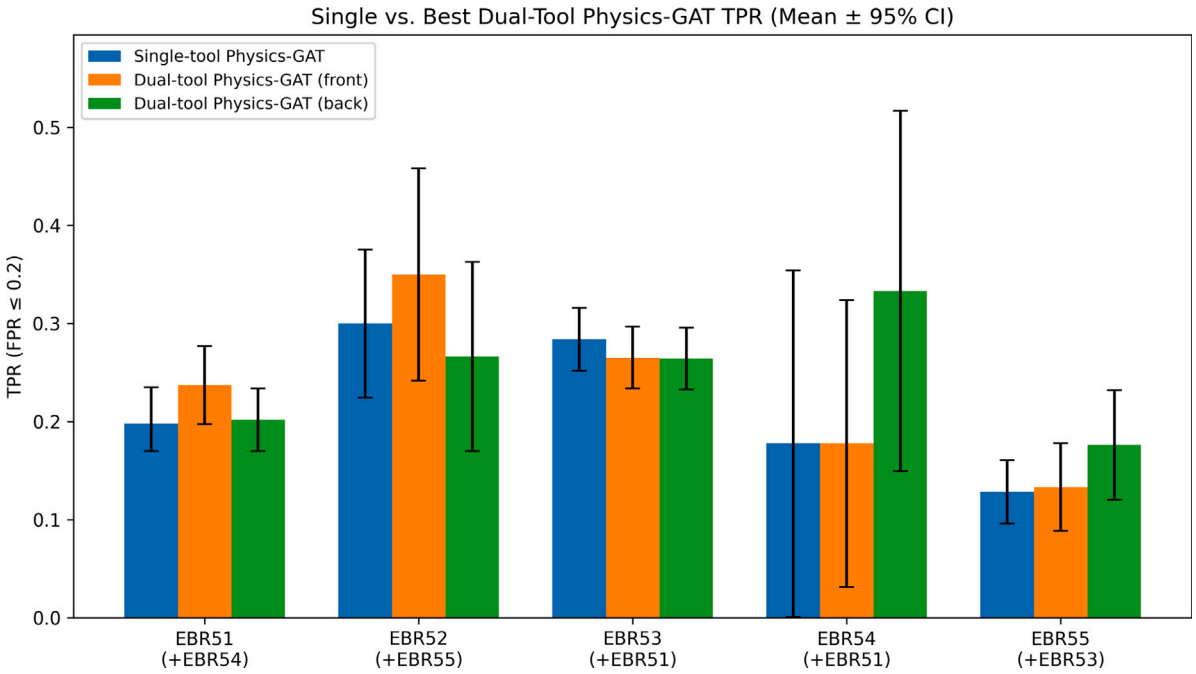

Fig. 8. Comparison between single-tool Physics-GAT and the best validation-selected dual-tool Physics-GAT configuration using the industrial metric of maximum TPR under the constraint FPR ≤ 0.2. Helper tools selected by validation are indicated below each target tool. Dual-tool training raises the best observed constrained-TPR on several tools, particularly EBR52 and EBR54.

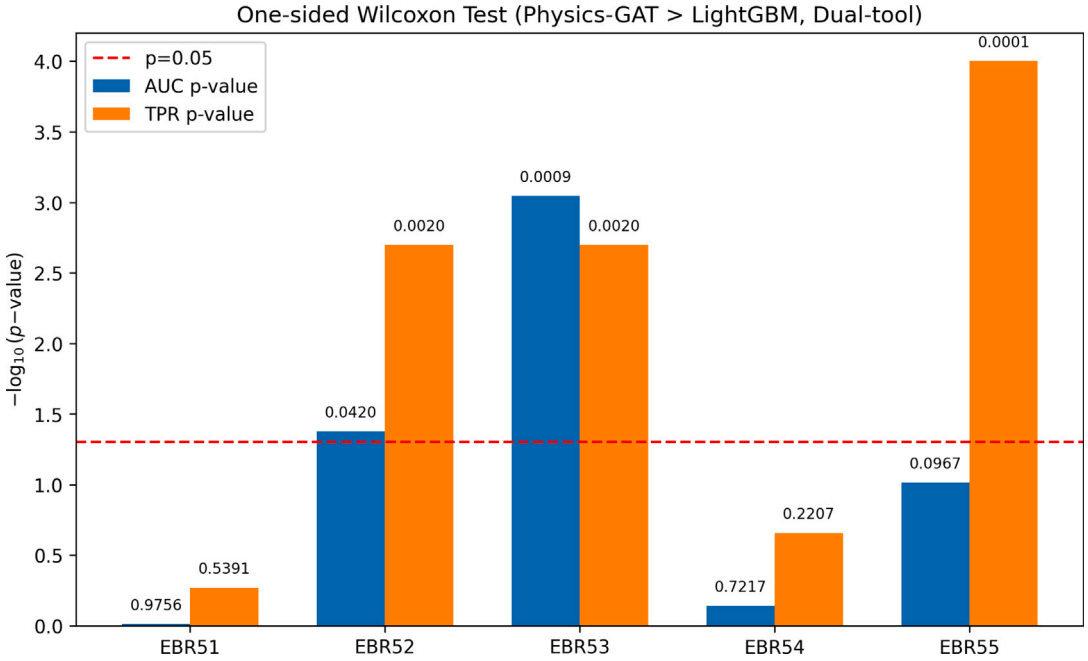

Fig. 9. One-sided Wilcoxon signed-rank test results for the hypotheses that the best dual-tool Physics-GAT outperforms the best dual-tool LightGBM benchmark in ROC–AUC and in TPR under FPR ≤ 0.2. Bars above the dashed line correspond to $p < 0.05$. Compared with the single-tool case, the dual-tool setting shows stronger evidence that Physics-GAT can outperform LightGBM, particularly on EBR52 and EBR53 in ROC–AUC and on EBR52, EBR53, and EBR55 in constrained TPR.

these results support the main hypothesis of this work: the physics-structured GAT is most promising in the multi-tool regime, where it shows stronger and more frequent advantages over the LightGBM benchmark, especially in the low-FPR operating region that is most relevant for industrial fault detection.

An essential comparison is whether the cross-tool meta-head GNN models—which combine the EBR polishing backbones with the NXE or G602 dry-etch backbones—can outperform the strong single-tool LightGBM benchmarks. Looking at the ROC–AUC values (Fig. 10), both the EBR52 and EBR53 meta-head GNN models successfully outperform their respective LightGBM baselines. Interestingly, whether the G602-assisted or the NXE-assisted meta-head GNN performs better depends heavily on whether noise levels or training data size is more critical for that specific target tool. For EBR53, noise level has a more significant impact on model performance because EBR53 already contains the largest dataset among all the EBR tools. Consequently, introducing a

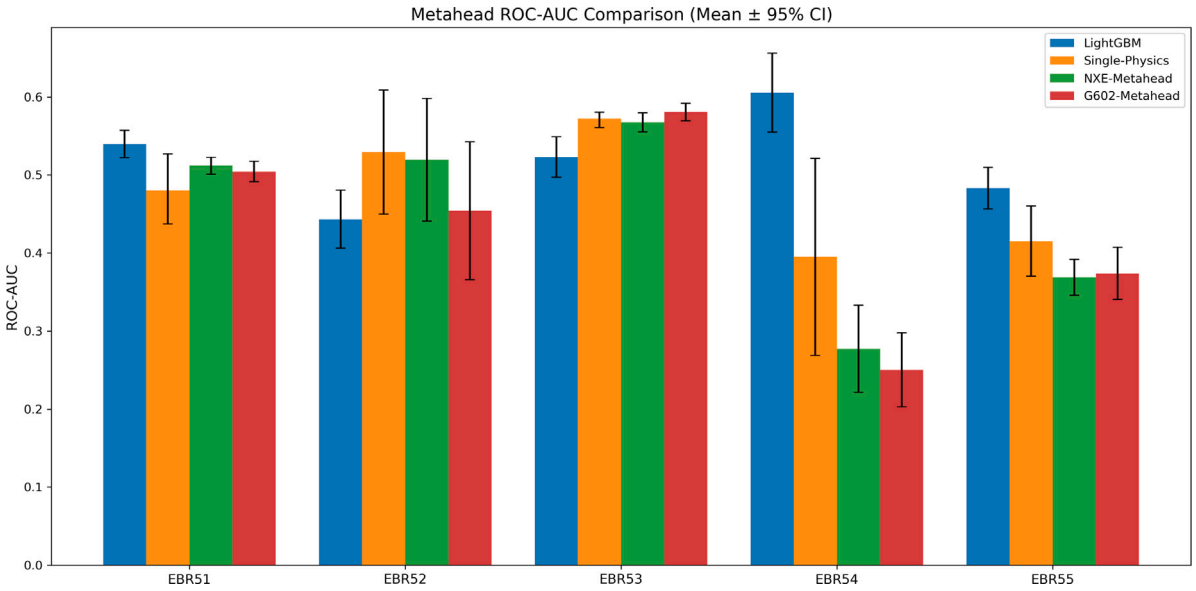

Fig. 10. Comparison between single-tool Physics-GAT, LightGBM baseline and the G602/NXE Metahead Physics-GAT configuration for each target tool. Metahead Physics-GAT improve on ROC–AUC value on several tools, especially EBR52, and EBR53.

cross-tool dataset with lower noise levels, such as the G602 dry-etch family, helps stabilize GNN training and yields a superior meta-head representation. As for EBR52, its data volume is only about 60% of that of EBR53. For this tool, increasing the training data size is a far more critical factor; therefore, leveraging the NXE dry-etch family, which provides a very large dataset, helps improve the model's representation more effectively than the quieter but smaller G602 dataset. In contrast, for tools other than EBR52 and EBR53 (namely EBR51, EBR54, and EBR55), the meta-head GNN models are unable to outperform the LightGBM baselines in ROC–AUC. This is especially apparent for EBR54, which exhibits the highest baseline performance among all polishing tools. We suspect the primary reasons for this limitation on EBR54 are either physical misclassification in the target labels or the possibility that most of the raw sensor data does not fit cleanly into the physical groups defined during the preprocessing stage.

Observing from Fig. 11, the constrained True Positive Rate (TPR under the constraint FPR <= 0.2) values mostly follow the same performance trends as the ROC–AUC, with a major exception observed on EBR55. Because EBR55 has the smallest dataset volume among all the EBR machines, its single-tool baseline performance is highly unstable and unacceptable for industrial deployment. Consequently, the cross-tool meta-head GNN introduces substantial performance improvements on EBR55 in almost all evaluation cases. This pronounced improvement is not reflected in the ROC–AUC metric; because AUC is a global curve-area metric, it does not isolate or evaluate performance looking solely at the accuracy of negative sample predictions under high-imbalance constraints, which are highly impacted by data scarcity in the low-FPR operating region.

7.3. Metric selection & PR-AUC vs. ROC-AUC baseline analysis

In high-volume semiconductor manufacturing with extreme class imbalance (a positive failure rate of approximately 1%–2%), selecting appropriate evaluation metrics requires balancing operational sensitivity with cross-tool comparability. While Precision–Recall AUC (PR-AUC) is frequently discussed for imbalanced datasets, comparing raw PR-AUC scores directly across different process tool fleets introduces substantial ambiguity. Unlike the constant 0.5 baseline of standard ROC–AUC, the uncalibrated baseline for PR-AUC equals the exact tool-specific positive prevalence rate ($N/(P+N) \approx 0.015$). Consequently, raw PR-AUC values cannot provide a normalized basis for cross-tool performance comparison without explicitly accounting for fleet-specific baseline variations.

To establish a standardized comparison, Fig. 12 presents a side-by-side evaluation of normalized ROC–AUC and baseline-adjusted PR-AUC for the single-tool Physics-GAT configuration across process runs. As shown in the empirical results, PR-AUC behavior is highly case-dependent—fluctuating significantly based on minor variations in tool-specific fault prevalence and baseline shifts. Because PR-AUC is so sensitive to these tool-dependent prevalence variations, relying on it as a primary metric can yield inconsistent interpretations across different process chambers. Therefore, we conclude that ROC–AUC and partial ROC–AUC ($pAUC_{FPR \leq 0.2}$) remain the most suitable and robust metrics for our setup, evaluating true positive rate improvements strictly within the operationally acceptable low false-alarm regime while maintaining standardization across distinct tool fleets.

7.4. Tabular model performance

Following the evaluation of the Physics-GAT, we investigated two specialized deep learning architectures designed specifically for tabular data: TabR and TabM. These models were selected to determine if deep learning architectures inherently optimized for tabular structures could provide a more robust alternative to the gradient-boosting baseline.

As demonstrated in Figs. 13 and 14, across the majority of the EBR tools, a consistent performance hierarchy emerges: while the LightGBM baseline often remains the superior model, TabR consistently outperforms TabM in terms of TPR value (FPR $\leq$ 0.2). Because TabR is an attention-based model while TabM is basically an ensemble of FNN models, this result aligns with the observations from the previous section, where the attention-driven Physics-GAT also demonstrated superior potential over standard tabular approaches. The exception to this trend is EBR53, which stands as the only tool where both the TabR and TabM models achieved better TPR and ROC–AUC results than the LightGBM baseline. For the other tools, although the TAB-optimized deep learning models do not surpass the benchmark, the relative strength of TabR over TabM suggests it is more effective at extracting features from this specific tabular dataset as in manufacturing the prior failures always act as strong reference for the near future.

To statistically validate these trends, one-sided Wilcoxon p-value calculations were performed for TabR across the five tools. The results confirm that LightGBM is significantly better than TabR on all

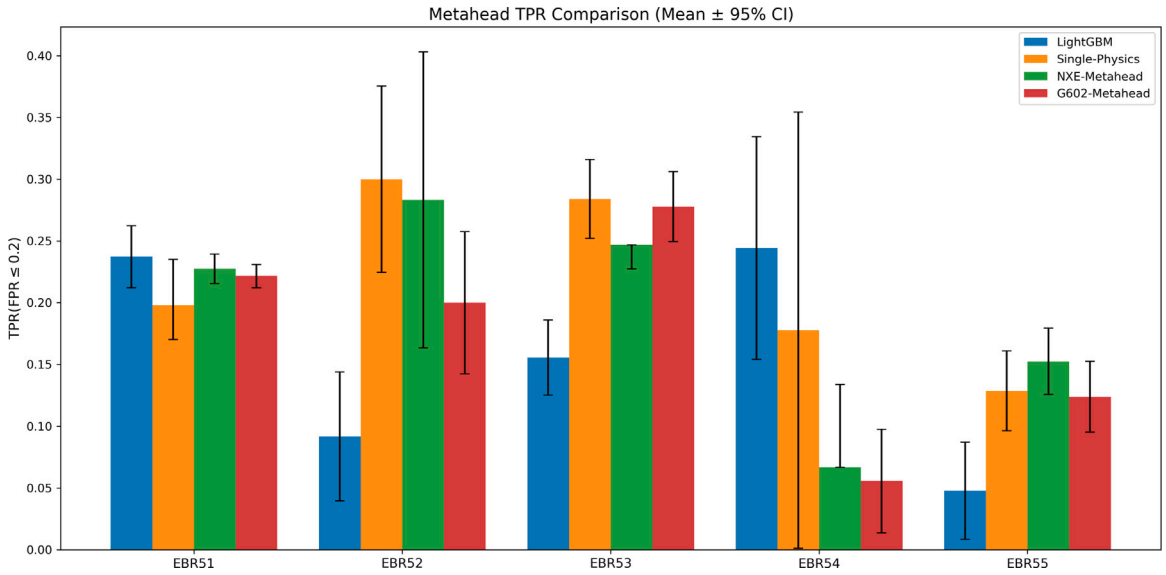

Fig. 11. Comparison between single-tool Physics-GAT, LightGBM baseline and the G602/NXE Metahead Physics-GAT configuration for each target tool. Metahead Physics-GAT improve on TPR value on several tools, especially EBR52, EBR53, and EBR55.

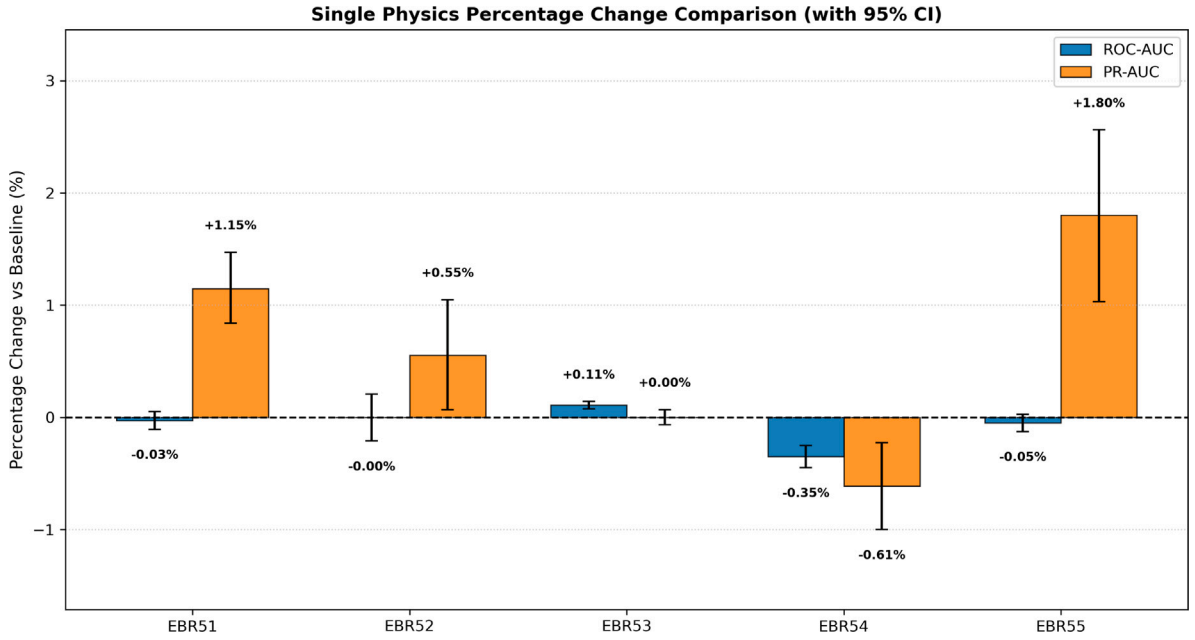

Fig. 12. Comparison of normalized ROC–AUC and baseline-adjusted PR-AUC for the single-tool Physics-GAT architecture. Because raw PR-AUC baselines vary with tool-specific defect prevalence rates (≈0.015), normalized metric representations demonstrate why partial ROC–AUC ($pAUC_{FPR \leq 0.2}$) provides a more standardized evaluation baseline across distinct tool fleets.

tools except for EBR53. This anomaly likely stems from the fact that EBR53 possesses a significantly higher fail rate than its counterparts, providing a richer density of "fail" samples that allows the TAB models more room for optimization and improvement. Ultimately, while these architectures are designed for tabular data, the binary nature of the wafer pass/fail problem and the class imbalance inherent in most tools means they do not consistently provide a "silver bullet" solution over optimized gradient boosting, though TabR shows the most promise of the two.

The statistical significance of these performance gains is further clarified by the *p*-value comparisons of the single-tool TabM and TabR models against the LightGBM baseline under the respective one-sided Wilcoxon signed-rank test hypotheses in Fig. 15. When utilizing single-tool training, we can only be certain about the statistically supported improvements on the EBR53 tool. For TabR, the improvement over the baseline is statistically significant on EBR53 in terms of ROC–AUC ($p < 0.05$). For TabM, the improvement on EBR53 is statistically supported in terms of both ROC–AUC and the constrained TPR metric ($p < 0.05$). The lack of significant movement on other tools, particularly on EBR54, is noteworthy. After looking into the data itself, we found that because EBR54 possesses a relatively low variety in terms of recipe types and process categories, the underlying data distribution is less complex. In such scenarios, the LightGBM baseline is already highly efficient at capturing the available signal, making it exceptionally difficult for a more complex deep learning model like TabM or TabR to provide a meaningful performance surplus. This suggests that while pooled training can help deep models overcome data scarcity, the ultimate ceiling of single-tool tabular deep learning models remains heavily constrained

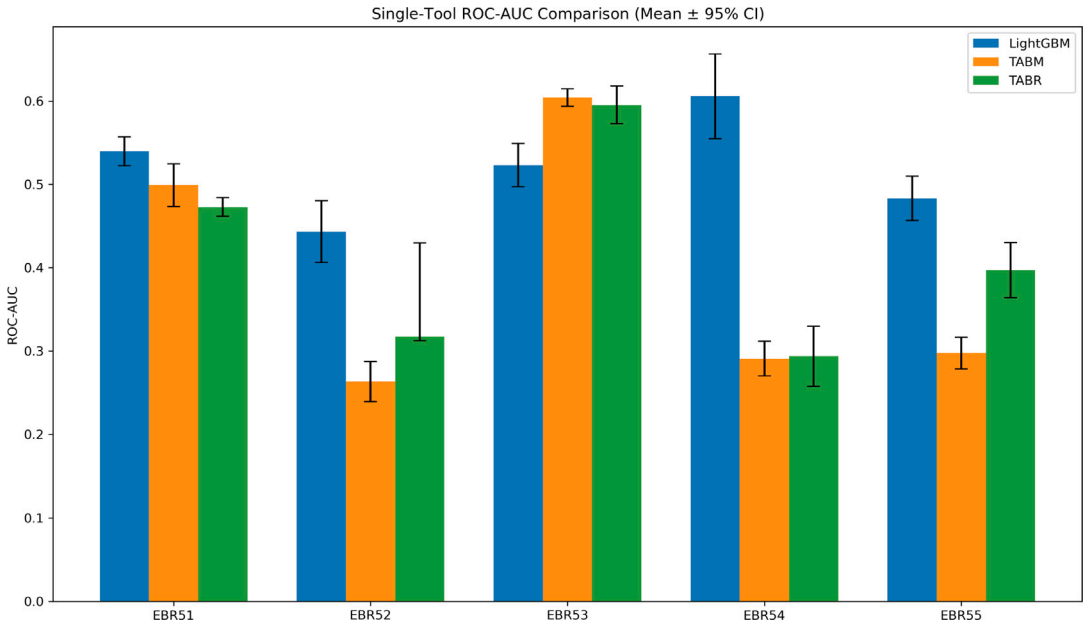

Fig. 13. Comparison based on AUC values between single-tool TabM, TabR and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. Tab model has better performance on data with higher fail rate like EBR53.

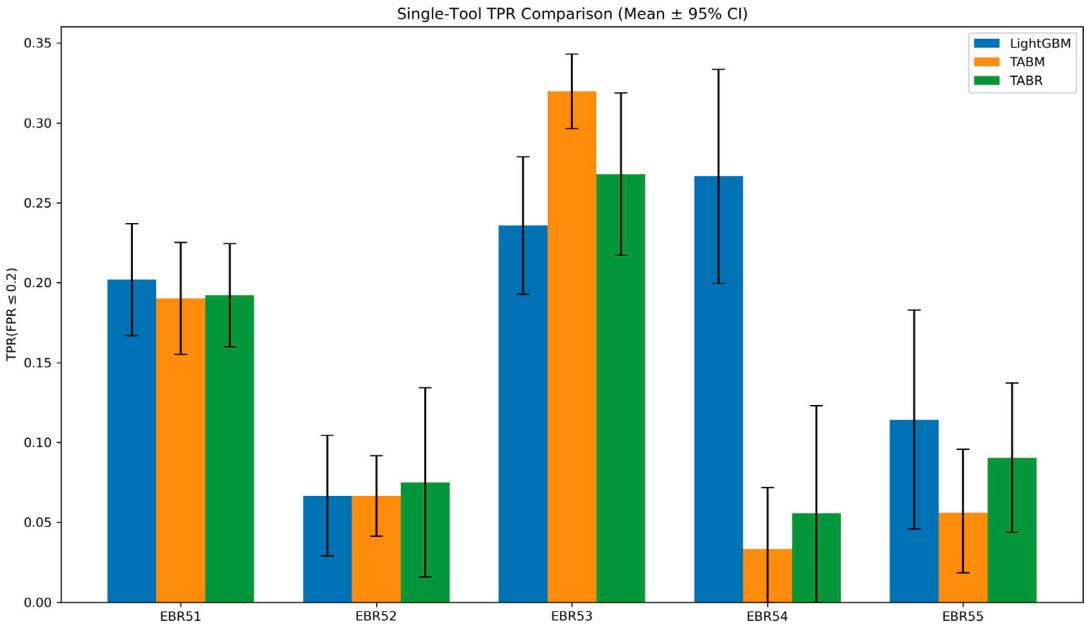

Fig. 14. Comparison based on TPR values between single-tool TabM, TabR and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. Tab model has better performance on data with higher fail rate like EBR53.

by the inherent operational diversity and failure mechanisms of the specific tool in question.

Regarding the evaluation of the hybrid TabM-LightGBM model in Figs. 16 and 17, the ROC–AUC performance of almost all tools surpasses that of the standalone single-tool TabM model, with the sole exception of EBR53. For EBR53, both the single and hybrid TabM configurations outperform the LightGBM baseline, whereas for all other tools, neither setup is able to exceed the baseline. Notably, EBR53 consistently outperforms the LightGBM baseline on the ROC–AUC score across most model variations. This unique behavior is likely due to the failure rate distribution across the fleet: although all tools exhibit very low failure rates, EBR53 has the largest failure rate (almost twice that of the other machines), which provides a larger optimization window and more room for performance improvement. When analyzing the constrained TPR metric of the hybrid TabM model, most tools (except for EBR54) fail to outperform the standalone single-tool TabM. This outcome is likely explained by the fact that the baseline TPR of EBR54 on the single TabM model is the lowest among all evaluated tools, making it the easiest target to outperform. Furthermore, both EBR53 and EBR55 in the hybrid TabM configuration successfully outperform the standalone LightGBM baseline, even though they do not exceed the performance of their standalone single-tool TabM counterparts.

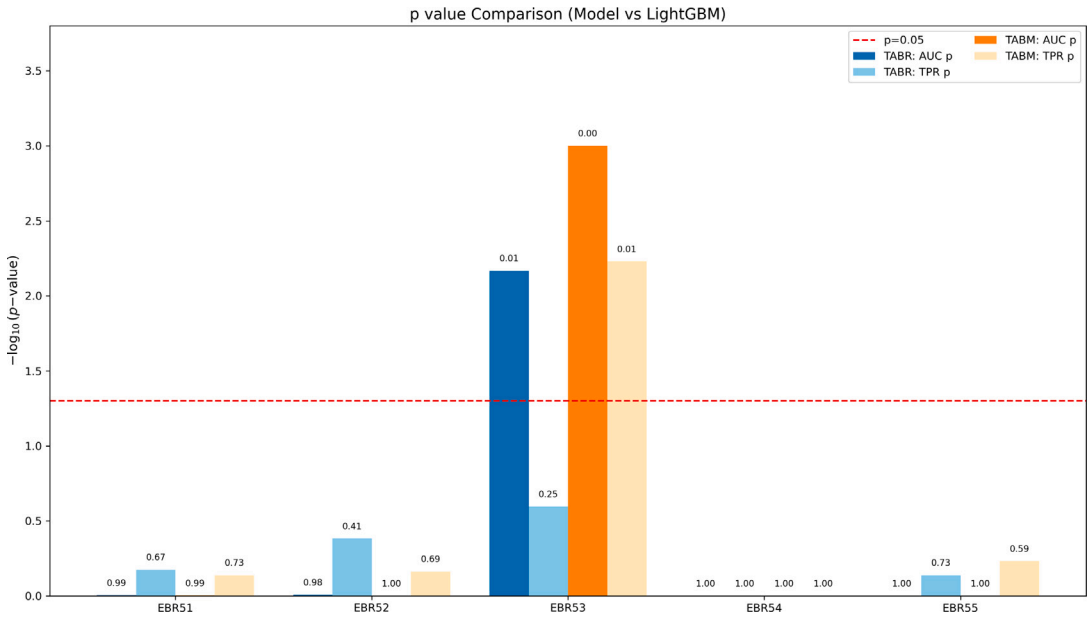

Fig. 15. One-sided Wilcoxon signed-rank test results for the hypotheses that the single-tool TabM and TabR outperforms the single-tool LightGBM benchmark in ROC–AUC and in TPR under FPR ≤ 0.2. Bars above the dashed line correspond to $p < 0.05$. We can only be certain about TabR outperforming LightGBM for EBR53 in terms of AUC and TabM in terms of both AUC and TPR.

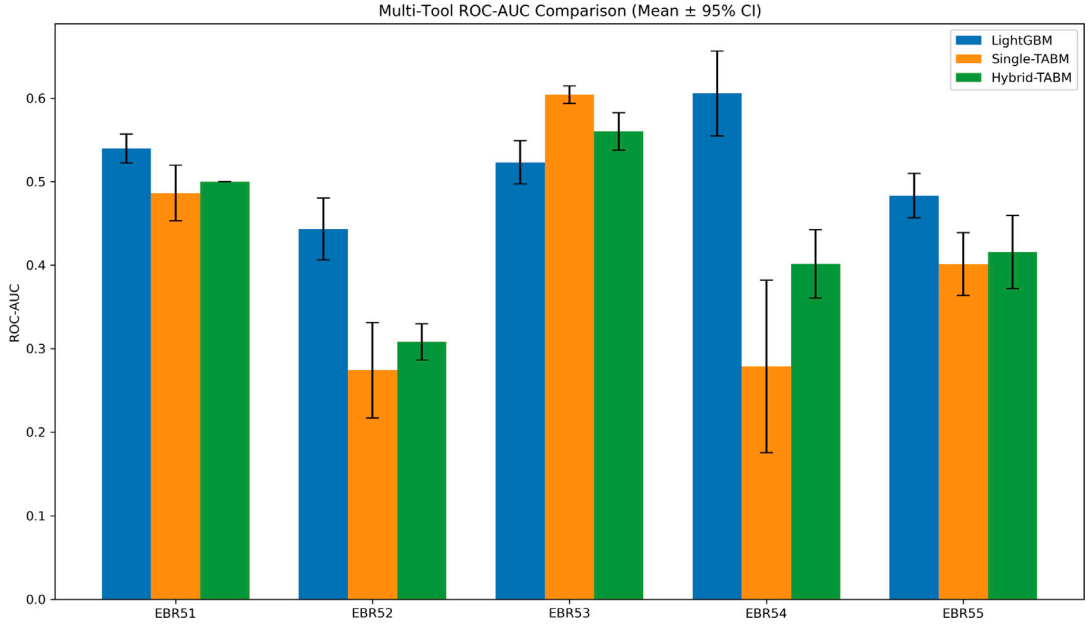

Fig. 16. Comparison based on AUC values between single-tool TabM, Hybrid TabM and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. TAB model has better performance on data with higher fail rate like EBR53.

The evaluation of the hybrid TabR-LightGBM model reveals a highly similar trend to the hybrid TabM model, with the primary difference occurring in the TPR of EBR54 in Figs. 18 and 19. In this case, the TPR of the hybrid TabR model on EBR54 does not yield a significant improvement compared to the standalone single-tool TabR. This lack of performance gain is suspected to stem from the high noise levels inherent to this specific tool. Compounding representation noise in a hybrid tree framework can be counterproductive, particularly for a tool like EBR54 that already possesses the largest quantity of useful, high-quality features among all tools in the fleet, thereby rendering additional feature stacking less effective. Table 4 summarizes the results of each model evaluated in this work with the emphasis on the advantages and disadvantages. Based on the findings reported on Table 4, the following recommendations can be considered in applying the evaluated methods and models with respect to tool data processing, size, failure history and operational regime complexity:

- **Analyze Model Generalization and Probability Calibration:** Evaluate empirical learning and loss curves immediately following baseline execution to diagnose high bias (underfitting), high variance (overfitting), or severe miscalibration (prognostic overconfidence). These diagnostics identify precisely where to inject targeted data preprocessing, particularly when high post-preprocessing noise requires auditing whether the available failure density provides a sufficient signal-to-noise ratio for effective learning.

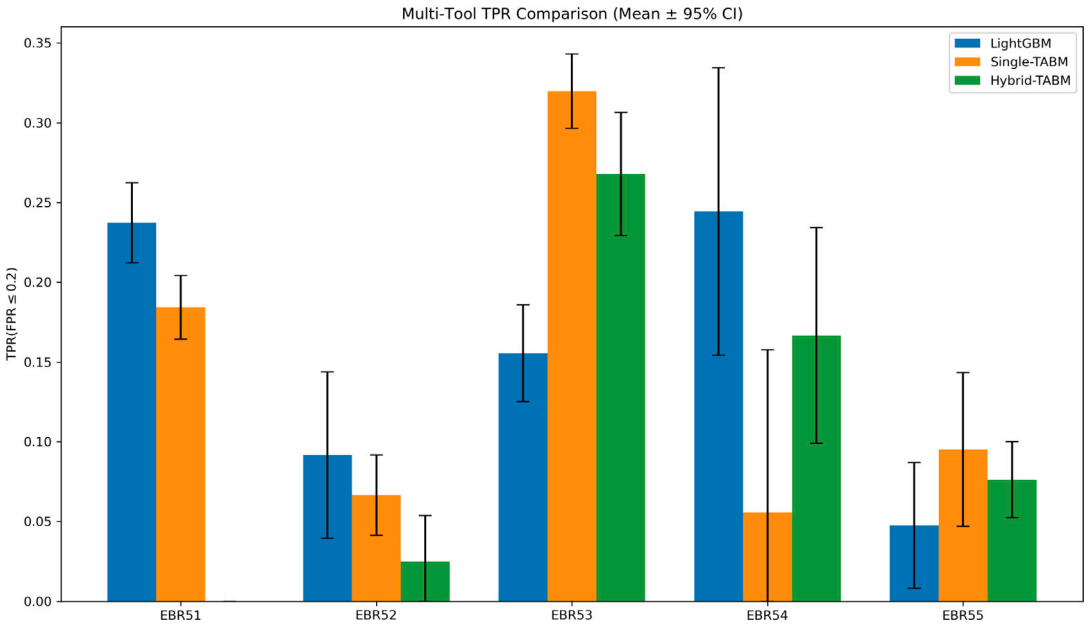

Fig. 17. Comparison based on TPR values between single-tool TabM, Hybrid TabM and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. TAB model has better performance on data with higher fail rate like EBR53.

- **Transition Data-Scarce Tools to Multi-Tool and Cross-Machine Graph Architectures:** Utilize physics-structured graph networks and multi-source representation learning to pool cross-tool data and distinct process backbones, which stabilizes volatile standalone metrics and improves performance ceilings for tools with low failure rates and limited data.
- **Deploy Retrieval-Augmented Tabular Models on High-Informed, High-Failure Regimes:** Leverage nearest-neighbor retrieval mechanisms on tools with richer failure histories to explicitly exploit recurring historical signatures and expand feature variety.
- **Implement Context-Aware Gating and Hybrid Tree-Neural Frameworks for Complex Regimes:** Route complex or noise-prone operations through dynamic, context-dependent routing rules rather than rigid model averaging, and use frozen deep neural representations to train gradient-boosted decision tree heads.

7.5. Context-aware gated ensemble and meta-learner performance

Finally, we evaluate the performance of the context-aware gated ensemble (the meta-learner), which dynamically weights predictions from the LightGBM, TabM, and TabR base models. The comparisons between the standalone LightGBM baseline and the gated meta-learner in terms of both ROC–AUC and TPR are illustrated in Figs. 20 and 21. However, the computational cost of the context-aware gated ensemble is exceptionally high because the gating network must dynamically calculate weights for each constituent model within the ensemble for every individual run. Consequently, to manage this heavy overhead, we restricted our meta-learner experiments to a single seed per tool, rather than the ten random seeds evaluated for the standalone tabular models.

Similar to the trends observed in the standalone TabM and TabR hybrid configurations, the meta-learner does not yield systemic performance improvements over the pure LightGBM baseline for most tools, with the sole exception of EBR53. For EBR53, the meta-learner successfully improves predictive performance over the baseline, benefiting from the fact that EBR53 possesses the highest failure density across the entire fleet and thus provides sufficient signal for multi-model optimization. Conversely, EBR54 is highly sensitive to model variations and experiences the most severe degradation in both ROC–AUC and constrained TPR when processed through the meta-learner. This performance drop is consistent with the standalone TabM and TabR behavior, and is largely driven by the intrinsic attention/gating mechanism of the meta-learner, which distributes model reliance across three distinct architectures whose individual tabular predictions may carry conflicting representation noise on a tool with low operational variety. These results are highly consistent with our previous findings which the deep tabular architectures (TabM and TabR) fail to consistently outperform the optimized LightGBM baseline in standalone single-tool settings, combining them via a gated meta-learner does not produce a stronger decision boundary than the pure, unmixed LightGBM baseline for the majority of tools.

8. Conclusion

This work developed and evaluated a physics-structured graph attention network (Physics-GAT) framework for pass/fail prediction in semiconductor manufacturing using real industrial tool data. Starting from AI-ready preprocessing and physics-informed feature generation, we established LightGBM as a strong benchmark model and then introduced a graph-based architecture that organizes sensor statistics according to physically meaningful tool subsystems while incorporating categorical production context through learned embeddings. The proposed framework was evaluated in both single-tool and multi-tool training settings under a temporally separated test protocol, with repeated random train/validation splits to provide statistically more reliable mean performance and confidence intervals. In addition to the Physics-GAT, this work also evaluated TabM and TabR as modern tabular deep-learning benchmarks and introduced a context-aware gated ensemble framework for combining complementary predictions from LightGBM, TabM, and TabR.

The results demonstrate that LightGBM remains a highly competitive benchmark, especially in the single-tool setting, but the proposed Physics-GAT demonstrates clear potential when richer structural information is exploited. In particular, the graph model demonstrated stronger advantages in the dual-tool setting, indicating that subsystem-level relational learning becomes more effective when additional cross-tool data are available. This suggests that the proposed method is able to capture transferable physical interactions that are not fully represented by flattened tabular features alone. In addition to predictive

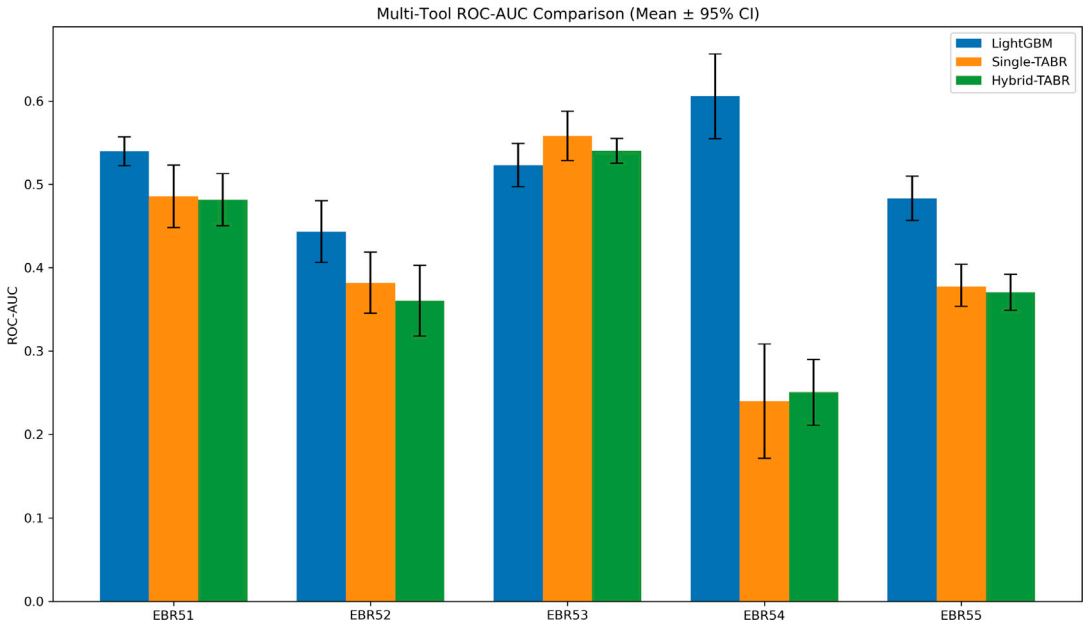

Fig. 18. Comparison based on AUC values between single-tool TabR, Hybrid TabR and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. Tab model has better performance on data with higher fail rate like EBR53.

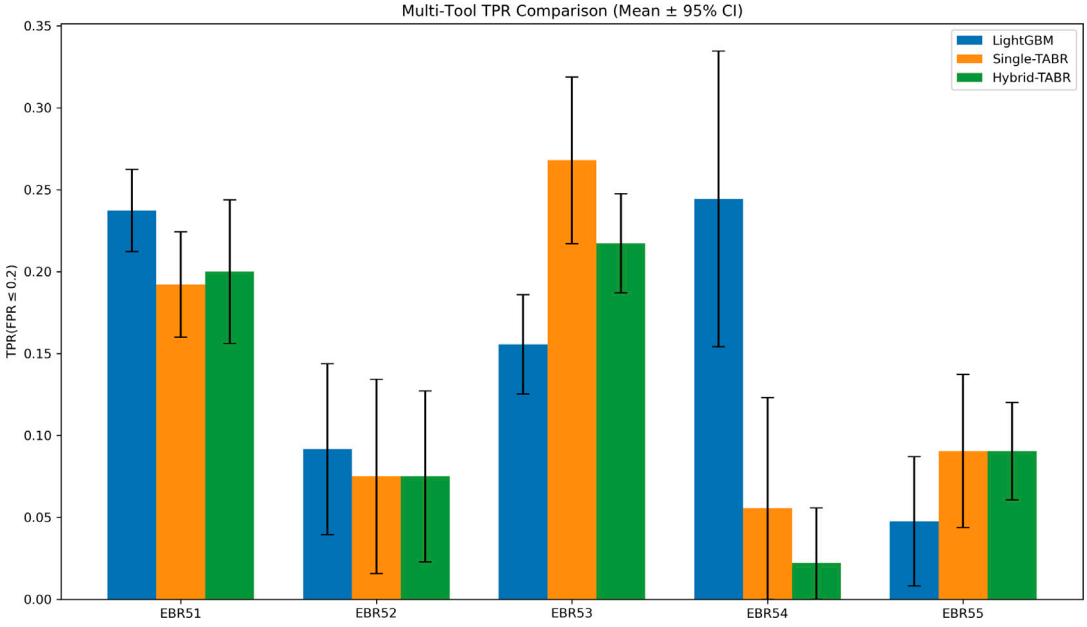

Fig. 19. Comparison based on TPR values between single-tool TabR, Hybrid TabR and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. Tab model has better performance on data with higher fail rate like EBR53.

performance, the Physics-GAT provides a natural pathway toward improved interpretability through within-group attention, subsystem-level aggregation, and cross-group interaction analysis, which are valuable properties for industrial fault analysis and engineering diagnosis.

The TabM and TabR results provide an additional perspective on the role of modern deep learning for tabular semiconductor manufacturing data. Across most EBR tools, LightGBM remains stronger than the specialized tabular neural models, demonstrating that optimized gradient boosting is still difficult to replace under severe class imbalance and limited failure samples. However, TabR generally outperforms TabM in the constrained low-FPR operating metric, suggesting that retrieval-augmented prediction can extract useful local historical similarity from the processed tabular feature space. EBR53 is the main exception where both TabM and TabR exhibit better performance than the LightGBM benchmark, likely because the higher failure rate provides more informative abnormal examples for neural tabular optimization. The dual-tool TabM results also indicate that pooled training can improve the TPR of neural tabular models for most tools, although the improvements are not universal; in particular, the limited gains on EBR54 suggest that when recipe and process-category variety is low, LightGBM may already capture most of the available signal.

At the current stage, this work should be viewed as a strong proof-of-concept rather than a final deployment-ready system. Future work should include further improvement of the graph architecture, broader hyperparameter and structure optimization, and additional analysis of when and why multi-tool aggregation is beneficial. A particularly important next step is to evaluate the interpretability outputs in a real industrial setting and determine whether the learned group-level

Table 4
Summary of investigated models and results.

Model	Model Class/Category	Key findings/Operational outcomes
Feedforward Neural Network (FNN)	Classical Deep Learning	Occasionally produces a better global ranking score (ROC-AUC) on individual tools (e.g., EBR53), but underperforms LightGBM in the strictly constrained low-FPR fault-detection regime.
LightGBM	Gradient Boosting Decision Tree (GBDT)	Proved to be the most reliable baseline globally, significantly outperforming FNNs in the low-FPR region. However, standalone performance falls short of the $\geq 80\%$ TPR industrial deployment threshold under severe class imbalance. Multi-tool aggregation significantly lifts this performance ceiling, though the gains remain highly dataset-dependent
Physics-GAT	Physics-Structured Graph Attention Network	Most effective in the multi-tool (dual-tool) regime, where it leverages additional data to stabilize subsystem relational patterns and significantly outperforms dual-tool LightGBM in constrained TPR on several tools (EBR52, EBR53, EBR55). It also provides unique diagnostic interpretability via internal attention pooling weights.
Cross-Machine GNN Metahead	Multi-Source Representation Learning	Successfully maps distinct process steps (polishing and dry-etching) to successfully outperform the LightGBM baseline in ROC–AUC on tools with adequate dataset size or manageable noise levels (EBR52, EBR53). It also drastically stabilizes the highly volatile standalone metrics of data-scarce tools (EBR55).
TabM	Tabular Deep Learning (Ensemble-style)	Generally falls short of optimized gradient boosting under severe class imbalance and limited failure densities, except on the tool with the richest failure rate (EBR53), where its parameter-efficient neural ensemble yields statistically significant improvements in both ROC–AUC and constrained TPR over LightGBM.
TabR	Tabular Deep Learning (Retrieval-augmented)	Consistently outperforms TabM in constrained TPR values because its nearest-neighbor retrieval mechanism exploits recurring historical failure signatures. Like TabM, it strictly beats LightGBM on the high-failure tool (EBR53), but remains capped on tools with low operational and recipe diversity (EBR54).
Hybrid Neural-Tree Framework	Hybrid Representation Learning	Successfully beats standalone TabM/TabR configurations across almost all tools by using frozen deep neural representations to expand feature variety. However, it became detrimental on specific tools (like EBR54) where compounding representation noise on a tool with high baseline noise levels degrades performance.
Context-Aware Gated Ensemble	Instance-Level Mixture of Experts	Successfully demonstrates that utilizing dynamic, context-dependent routing rules handles severe data constraints better than rigid uniform model averaging, allowing engineers to interpret average model reliance across changing recipe regimes.

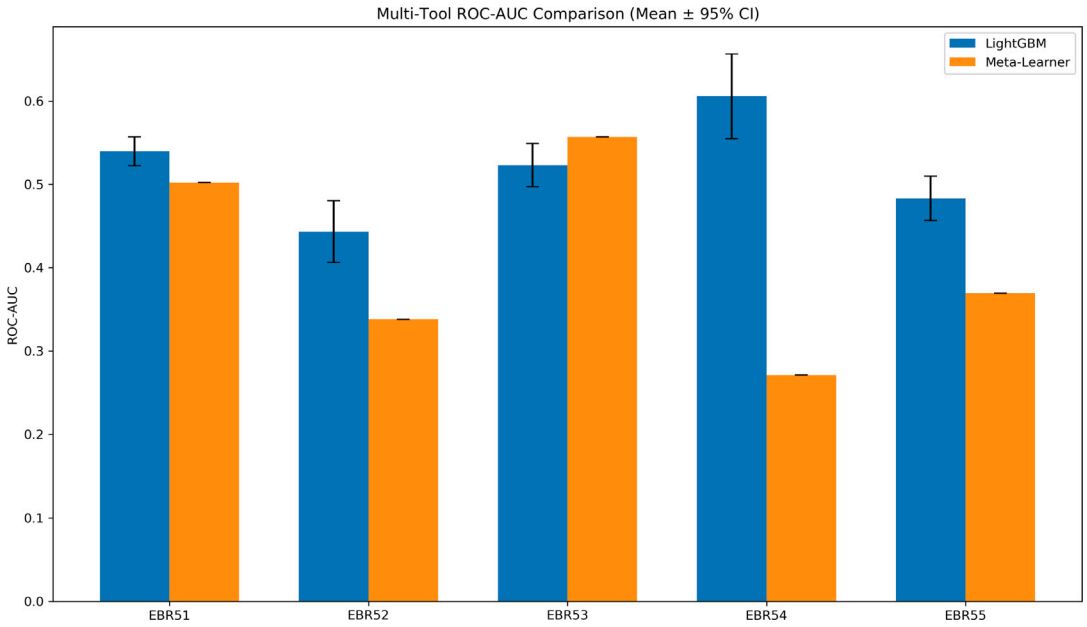

Fig. 20. Comparison based on AUC values between single-tool Meta-Learner and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. Meta-Learner model has better performance on data with higher fail rate like EBR53.

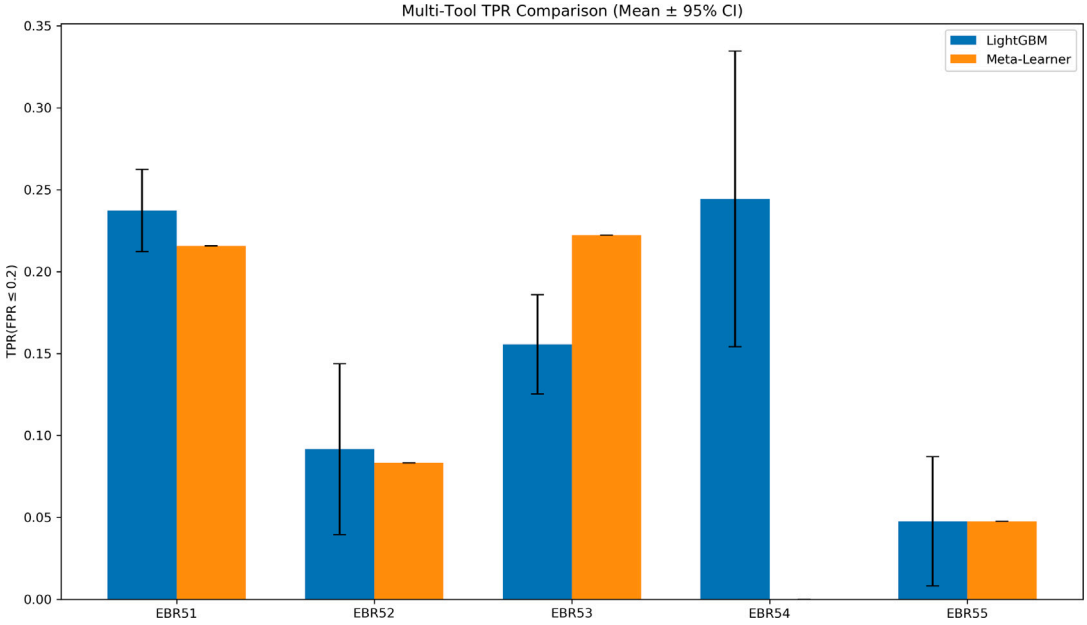

Fig. 21. Comparison based on TPR values between single-tool Meta-Learner and the LightGBM configuration for each target tool. Error bars denote the reported 95% confidence intervals. Meta-Learner model has better performance on data with higher fail rate like EBR53.

and sensor-level importance patterns provide actionable references for process engineers. Because no dedicated fault-history dataset is currently available, the explainability component cannot yet be directly validated against known root-cause records; however, this remains a promising direction for future industrial study. For the tabular models, future work should further investigate when retrieval-augmented learning is beneficial, how the retrieved neighbors from TabR can support engineering diagnosis, and how context-aware gated ensembles can combine LightGBM, TabM, and TabR more effectively across different tool, recipe, and process regimes.

Another important direction is to test robustness more rigorously through rolling temporal evaluation over multiple time windows before shadow testing in the real manufacturing environment. Such experiments would better reflect process drift, maintenance effects, and changing operating conditions over time. Combined with future industrial validation of the explainability outputs and further evaluation of complementary tabular neural models, these efforts can further clarify the practical value of physics-structured graph learning, retrieval-augmented tabular learning, and context-aware ensemble modeling for semiconductor manufacturing monitoring, soft sensing, and fault-detection support.

CRediT authorship contribution statement

Chun-Pei Lin: Writing – original draft, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Feiyang Ou:** Writing

– review & editing, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Yifei Wang:** Investigation. **Chao Zhang:** Writing – review & editing, Supervision, Methodology, Investigation. **Sthitie Bom:** Resources. **James F. Davis:** Writing – review & editing, Supervision, Methodology, Investigation, Conceptualization. **Panagiotis D. Christofides:** Writing – review & editing, Supervision, Methodology, Investigation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors gratefully acknowledge financial support from the U.S. Department of Energy, through the Office of Energy Efficiency and Renewable Energy (EERE), under the Advanced Manufacturing Office Award Number DE-EE0007613. Industrial data and support from Seagate Technology is gratefully acknowledged. This work used computational, and storage services associated with the Hoffman2 Shared Cluster provided by UCLA Office of Advanced Research Computing's Research Technology Group.

Disclaimer

This report was prepared as an account of a work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express, or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of the authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

References

Battaglia, P.W., Hamrick, J.B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al., 2018. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261.

Christofides, P.D., Davis, J.F., El-Farra, N.H., Clark, D., Harris, K.R.D., Gipson, J.N., 2007. Smart plant operations: Vision, progress and challenges. AIChE J. 53, 2734–2741.

Cover, T.M., Hart, P.E., 1967. Nearest neighbor pattern classification. IEEE Trans. Inform. Theory 13, 21–27.

Davis, J.F., et al., 2020. Cyberinfrastructure for the democratization of smart manufacturing. In: Smart Manufacturing. Elsevier.

Dreyfus, P.A., Psarommatis, F., May, G., Kiritsis, D., 2022. Virtual metrology as an approach for product quality estimation in Industry 4.0: a systematic review and integrative conceptual framework. Int. J. Prod. Res. 60, 742–765.

Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E., 2017. Neural message passing for quantum chemistry. In: Proceedings of International Conference on Machine Learning. Sydney, Australia, pp. 1263–1272.

Gonçalves, L., Subtil, A., Oliveira, M.R., De Zea Bermudez, P., 2014. ROC curve estimation: An overview. REVSTAT–Statistical J. 12, 1–20.

Gorishniy, Y., Kotelnikov, A., Babenko, A., 2025. TabM: Advancing tabular deep learning with parameter-efficient ensembling. In: Proceedings of International Conference on Learning Representations. http://dx.doi.org/10.48550/arXiv.2410.24210, arXiv:2410.24210.

Gorishniy, Y., Rubachev, I., Kartashev, N., Shlenskii, D., Kotelnikov, A., Babenko, A., 2024. TabR: Tabular deep learning meets nearest neighbors. In: The Twelfth International Conference on Learning Representations. URL: https://openreview.net/forum?id=rhgIgTSSxW.

Gorishniy, Y., Rubachev, I., Khrulkov, V., Babenko, A., 2021. Revisiting deep learning models for tabular data. In: Proceedings of Advances in Neural Information Processing Systems, vol. 34, pp. 18932–18943.

Jacobs, R.A., Jordan, M.I., Nowlan, S.J., Hinton, G.E., 1991. Adaptive mixtures of local experts. Neural Comput. 3, 79–87.

Jiang, Y., Yin, S., Dong, J., Kaynak, O., 2020. A review on soft sensors for monitoring, control, and optimization of industrial processes. IEEE Sensors J. 21, 12868–12881.

Kadlec, P., Gabrys, B., Strandt, S., 2009. Data-driven soft sensors in the process industry. Comput. Chem. Eng. 33, 795–814.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y., 2017. LightGBM: A highly efficient gradient boosting decision tree. In: Proceedings of Advances in Neural Information Processing Systems, vol. 30, pp. 3149–3157.

Ko, H., Lu, Y., Yang, Z., Ndiaye, N.Y., Witherell, P., 2023. A framework driven by physics-guided machine learning for process-structure-property causal analytics in additive manufacturing. J. Manuf. Syst. 67, 213–228.

Van der Laan, M.J., Polley, E.C., Hubbard, A.E., 2007. Super learner. U.C. Berkeley Division of Biostatistics Working Paper Series. Working Paper 222.

Liu, Y., Shen, B., Wong, D.S.-H., Jia, M., Yao, Y., 2026. Explainable neural network meets graph neural network: Recent advances in process fault detection and diagnosis. Comput. Chem. Eng. 226, 109528.

Maitra, V., Su, Y., Shi, J., 2024. Virtual metrology in semiconductor manufacturing: Current status and future prospects. Expert Syst. Appl. 249, 123559.

Meng, C., Griesemer, S., Cao, D., Seo, S., Liu, Y., et al., 2025. When physics meets machine learning: a survey of physics-informed machine learning. Mach. Learn. Comput. Sci. Eng. 1, 20.

Ou, F., Suherman, J., Zhang, C., Wang, H., Bom, S., Davis, J.F., Christofides, P.D., 2025. Industrial multi-machine data aggregation, AI-ready data preparation, and machine learning for virtual metrology in semiconductor wafer and slider production. Digit. Chem. Eng. 15, 100242.

Ou, F., Wang, H., Zhang, C., Tom, M., Bom, S., Davis, J.F., Christofides, P.D., 2024. Industrial data-driven machine learning soft sensing for optimal operation of etching tools. Digit. Chem. Eng. 13, 100195.

Sun, Q., Ge, Z., 2021. A survey on deep learning for data-driven soft sensors. IEEE Trans. Ind. Informatics 17, 5853–5866.

Suthar, K., Shah, D., Wang, J., He, Q.P., 2019. Next-generation virtual metrology for semiconductor manufacturing: A feature-based framework. Comput. Chem. Eng. 127, 140–149.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Proceedings of Advances in Neural Information Processing Systems, vol. 30, Long Beach, CA, USA.

Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y., 2018. Graph attention networks. In: Proceedings of International Conference on Learning Representations. arXiv:1710.10903.

Wen, Y., Tran, D., Ba, J., 2020. BatchEnsemble: An alternative approach to efficient ensemble and lifelong learning. In: Proceedings of International Conference on Learning Representations. arXiv:2002.06715.

Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biom. Bull. 1, 80–83.

Wolpert, D.H., 1992. Stacked generalization. Neural Netw. 5 (2), 241–259.

Wu, X., Zhang, C., Du, W., 2021. An analysis on the crisis of "chips shortage" in automobile industry——Based on the double influence of COVID-19 and trade friction. J. Phys.: Conf. Ser. 1971, 012100.

Wu, Y., et al., 2024. Physics-informed machine learning: A comprehensive review on applications in anomaly detection and condition monitoring. Expert Syst. Appl. 255, 124678.

Zhang, C., Yella, J., Huang, Y., Qian, X., Petrov, S., Rzhetsky, A., Bom, S., 2021. Soft sensing transformer: hundreds of sensors are worth a single word. In: Proceedings of IEEE International Conference on Big Data. Orlando, Florida, pp. 1999–2008.

Zhou, J., Cui, G., Hu, S., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M., 2020. Graph neural networks: A review of methods and applications. AI Open 1, 57–81.