

A Reinforcement Learning Framework for Good Die in Bad Neighborhood Analysis

Mohammad Ershad Shaik*, Abhishek Mishra†, Nagarajan Kandasamy† and Nur A. Touba*

*Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX 78712, USA

†Electrical and Computer Engineering Department, Drexel University, Philadelphia, PA 19104, USA

*mohammad.ershada@utexas.edu, †am4862@drexel.edu, †nk78@drexel.edu, *touba@ece.utexas.edu

Abstract—Good-Die-in-Bad-Neighborhood (GDBN) analysis is a critical challenge in semiconductor manufacturing, where overly aggressive rejection reduces yield, while lenient acceptance increases test escapes and outgoing defective parts per million (DPPM). This asymmetric trade-off creates a multi-objective optimization problem spanning defect coverage, yield preservation, and return-material-authorization cost, often beyond the reach of conventional gradient-based methods. In this work, we employ reinforcement learning to develop an attention-based Deep Q-Network (DQN) framework tailored for GDBN-driven decision making. The DQN agent learns an optimal die-level screening policy from local wafer patches along with numerical test parametric data, optimizing actions that maximize cumulative long-term reward. By incorporating an attention mechanism, our model captures neighborhood-aware spatial dependencies across dies, enabling context-sensitive decision-making that balances yield and quality. We evaluated our method on the publicly available WM-811K wafer dataset, demonstrating substantial improvements in DPPM reduction and yield-cost tradeoffs compared to existing approaches. The results demonstrate that reinforcement learning provides a scalable and effective solution for adaptive defect screening in high-volume semiconductor test environments.

Index Terms—Semiconductor device testing, good-die-in-bad-neighborhood, silent data corruption, reinforcement learning

I. INTRODUCTION

Semiconductor manufacturing using advanced technology nodes is increasingly constrained by variability, latent defects, and design marginalities that challenge conventional test strategies. Wafer-level testing and screening in the manufacturing flow play a vital role in minimizing the impact of such constraints. Testing at this level has a direct impact on both yield loss and DPPM observed during the system-level test. Wafer-level testing is more critical in system-on-wafer integration, where thousands of dies are interconnected through redistribution layers (RDL) to form large-scale information processing systems [1], [2]. Ensuring a known good die before adding RDL is essential, as even a single latent defect escaping test can potentially induce silent data corruption or degrade system performance [3]–[5]. Defective dies can block communication paths, inject silent errors, or trigger cascading failures, meaning the yield of the entire wafer system depends on the weakest die. Such failures not only compromise system reliability, but also waste the high cost of fabrication, since one defective die can force full-wafer scrap.

Although structural and functional tests achieve high fault coverage, they cannot always capture subtle defect modalities that manifest under neighborhood effects or delayed

stress conditions. A prominent way is the *Good-Die-in-Bad-Neighborhood (GDBN)* analysis, where a die that individually passes parametric and functional tests appears suspicious due to its spatial proximity to defective neighbors. GDBN analysis improves wafer-level adaptive testing by using spatial clustering to identify marginal dies that standard tests may miss. By screening out dies early, GDBN reduces latent test escapes and lowers overall manufacturing cost while safeguarding the reliability of system-on-wafer integration.

GDBN is deployed to further suppress downstream defective parts per million (DPPM). However, there is a limit on the yield loss that can be incurred due to false rejects, dictated by the product cost budget. Improper handling of GDBN can lead to excessive yield loss when good dies are rejected or to increased DPPM when marginal dies escape detection. The cost incurred with incorrect analysis is asymmetric, which complicates test analytics and has a significant impact on the return-material-authorization (RMA) cost. In addition, GDBN analysis must be completed in a short period of time before the wafer lot is sent to packaging and system-level testing to maximize factory throughput time [6]. Manual analysis and adjustment of screening limits is time-consuming and risks affecting throughput. This creates a multivariate optimization problem in test escapes, yield, & throughput time, motivating the need for policy-based learning to maximize cumulative manufacturing gain.

Supervised learning methods, particularly artificial neural networks (ANNs) and convolutional neural networks (CNNs), have been used for wafer map classification and GDBN [7]–[11]. ANNs and CNNs provide strong spatial feature learning, but are limited to static classification tasks and cannot capture the sequential dynamics of wafer screening, where each decision can influence the effective distribution of subsequent neighborhoods. They do not naturally encode the asymmetric costs of false accepts versus false rejects, which are critical in manufacturing economics. They also do not account for product-specific economic trade-offs, often resulting in either excessive yield sacrifice or insufficient test escape suppression.

Unlike supervised methods, *Reinforcement Learning (RL)* provides a framework in which agents learn to optimize by interacting with an environment and receiving feedback in the form of rewards [12]. Instead of relying on large volumes of labeled training data, RL agents improve their behavior through trial and error, guided by reward functions that can

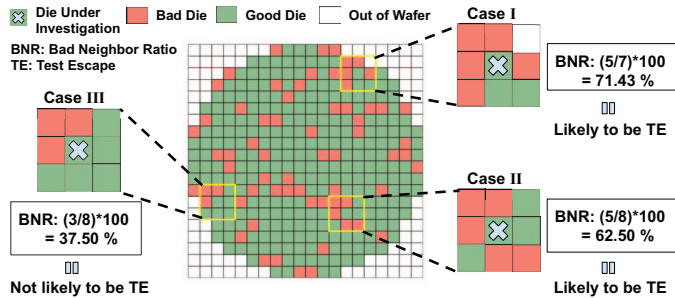


Fig. 1. Principle of Good-Die-in-Bad-Neighborhood (GDBN).

explicitly encode domain-specific objectives. This makes RL particularly attractive for complex decision-making tasks in semiconductor manufacturing, where trade-offs are inherently multi-objective and data labeling is expensive. One of the most widely used RL algorithms is Q-learning, which seeks to estimate the action-value function [13]. Classical Q-learning relies on a tabular representation, which becomes infeasible for large or continuous state spaces. To address this limitation, *Deep Q-Networks (DQNs)* extend Q-learning by introducing deep neural networks as powerful function approximators for the Q-function [14].

We develop an RL framework for GDBN analysis based on Deep Q-Learning and multimodal wafer maps and test data. The contributions of this work are summarized as follows:

- We formulate GDBN analysis as a reinforcement learning problem and design a framework tailored to wafer-level screening.
- We develop an attention-based DQN agent architecture that learns an optimal die-level screening policy from local wafer patches with numerical test parametric data, optimizing actions that maximize the catch of GDBN.
- We conduct experiments on open source real-world WM-811K [15] dataset to demonstrate the results with respect to existing approaches.

The paper is organized as follows. Section II provides the necessary background, while Section III details the proposed approach to formulate the problem. Section IV presents the experimental results, and Section V concludes the paper.

II. BACKGROUND

We introduce the GDBN methodology [9], [10], [16], [17], multi-modal learning [18] and the reinforcement learning study from the literature [12], [19], [20].

A. GDBN methodology

The basic idea of GDBN is captured in Fig. 1 in which *green squares* represent good dies (passing all tests), *red squares* represent bad dies (failing at least one test), *squares with a cross* denote the die under investigation (DUI) and *white squares* indicate regions outside the valid wafer area. To evaluate whether a DUI should be flagged as potentially defective, GDBN computes the *bad neighbor ratio* as $BNR = \frac{N_{\text{bad}}}{N_{\text{valid}}}$, where N_{bad} is the number of bad dies among the eight immediate neighbors of the DUI, and N_{valid} is the total number of valid (non-missing) neighboring dies.

TABLE I
NOTATION USED FOR THE GDBN PROBLEM.

Notation	Description
D	Number of defective dies regardless of test result
DC	Defect coverage (ratio of defects covered by testing)
P	Number of positives (dies that pass the test)
N	Number of negatives (dies that fail the test)
NN	Net negatives (dies in N with $SL \geq$ threshold)
L	Number of rejected dies out of the positives
TE	Number of test escapes (defective dies passing test)
CTE	Number of captured/covered test escapes
RTE	Number of remaining test escapes
Y	Yield ($P/(P + N)$)
YL	Yield loss ($L/(P + N)$)

Each zoomed-in region in Fig. 1 shows a DUI and its eight immediate neighbors, which are evaluated to calculate the BNR. For example, the DUI in the top right inset (CASE I) has a BNR of 71.43%, indicating a higher probability of latent defects and suggesting that the die is more likely to be a *test escape (TE)*. In contrast, the DUI highlighted in the left inset (CASE III) has a low BNR of 37.5%, indicating that it is not likely to be a TE. More interestingly, CASE II has a BNR of 62.5%, which is close to the *suspiciousness level (SL)* of 50%. This is also likely to be TE, but the decision to reject it is difficult, as there might be a large number of such dies and rejecting them all will result in a higher yield loss.

Previous studies have used metrics and notation in Table I such as the number of covered test escapes, DPPM, yield loss, and GAIN to assess model performance [9], [10], [17], [21]. Due to space limitations, we refer the reader to these studies for detailed derivation of metrics.

B. Multi-modal learning

Complementary information from multiple data modalities—images, text, audio, or video—can be integrated to improve model performance and generalization [22]–[24]. There are two primary strategies for fusing modalities: early fusion and late fusion. We adopt a late fusion strategy in our framework for the test escape detection task, integrating both wafer map images and numerical test data.

C. Reinforcement Learning

RL has been successfully applied in domains ranging from robotics to logistics, and more recently, chip design and manufacturing [25]–[29], providing a mathematical framework for sequential decision making under uncertainty, where an agent interacts with an environment to achieve a long-term objective [12]. Unlike supervised learning, which relies on fixed-label datasets, RL emphasizes *learning from interaction and delayed feedback*. At each discrete time step, the agent observes a state s_t , chooses an action a_t , and receives a scalar reward r_t . The environment then transitions to a new state s_{t+1} according to its underlying dynamics. The agent’s objective is to learn a policy $\pi(a|s)$ that maximizes the expected discounted cumulative reward

$$\pi^* = \arg \max_{\pi} \mathbb{E} \left[\sum_{t=0}^T \gamma^t r_t \right],$$

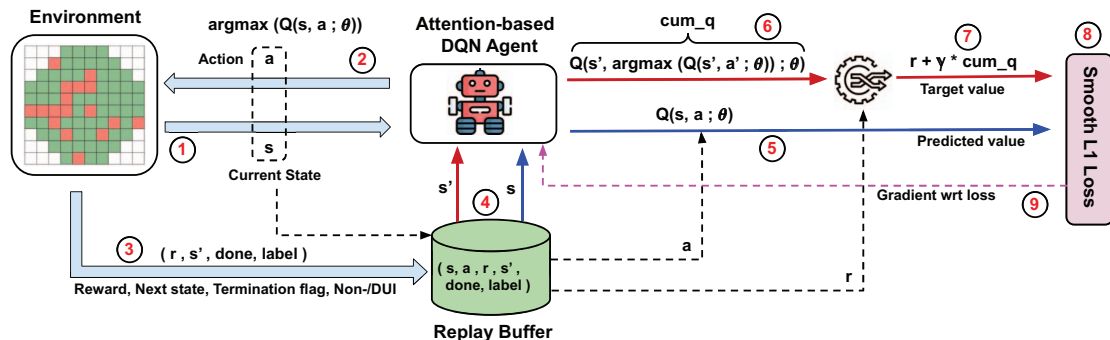


Fig. 2. The workflow for GDBN when posed as a reinforcement learning problem.

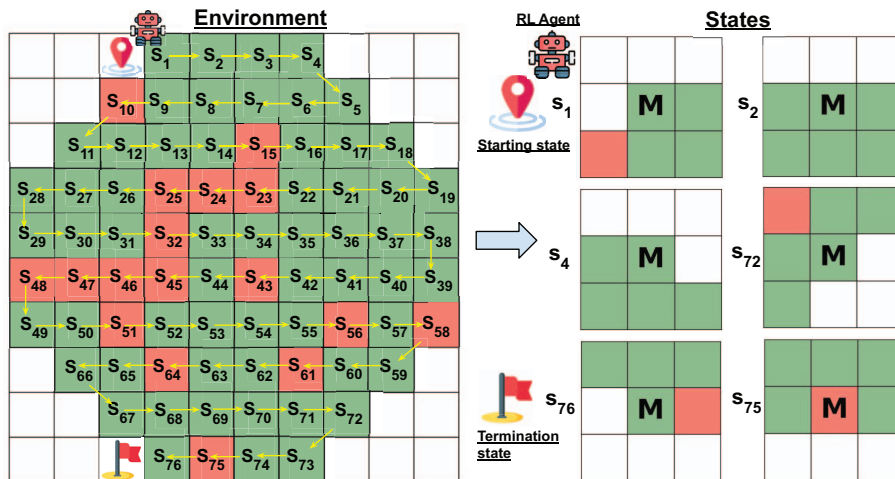


Fig. 3. The wafer map environment modeled as a Markov decision process (MDP). Each die, together with its eight neighbors and associated test parameters, defines a state S_i . The RL agent starts at s_1 and transitions sequentially across the wafer, receiving rewards for correct classifications of dies (DUI, good, bad). An episode starts at S_1 and terminates at the leftmost die in the bottom row (S_{76}), with cumulative reward reflecting the agent's performance.

where $\gamma \in (0, 1)$ is the discount factor that controls the trade-off between immediate and long-term rewards.

This sequential decision-making paradigm is commonly modeled as a Markov Decision Process, defined by the tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where \mathcal{S} denotes the state space, \mathcal{A} the action space, $P(s'|s, a)$ the transition probability, and $R(s, a)$ the reward function. The *optimal action-value function*, $Q^*(s, a)$, captures the maximum expected return achievable from state-action pair (s, a) . The agent follows an ϵ -greedy exploration strategy, balancing exploration with exploitation of learned estimates. At each time step, transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer and mini-batches are sampled to update the parameters. The learning objective is derived from the *Bellman optimality equation* [30]

$$Q^*(s, a) = \mathbb{E} \left[r + \gamma \max_{a'} Q^*(s', a') \mid s, a \right], \quad (1)$$

where r is the immediate reward, γ is the discount factor and s' is the next state. A Deep Q-Network *DQN* approximates the value function using deep neural networks [14]. Learning minimizes the temporal-difference between the predictions and the Bellman target.

III. PROBLEM FORMULATION

The GDBN problem is formulated as an RL task in which wafer-level test decisions are modeled as sequential interactions

between the environment and the RL agent, as shown in Fig. 2. This section describes our formulation in greater detail.

A. Environment and Episodes

Figure 3 shows the wafer map modeled as an RL environment where each die, also called Die Under Inspection (DUI), has a unique state S_i : it can be a good die (green) or a bad die (red). Each state is defined not only by the die itself but also by its eight immediate neighbors, which form a 3×3 local observation window, together with the corresponding seven numerical-test parametric data. At the beginning of an episode, the environment loads a wafer map and presents the first state S_1 to the agent. If the agent selects the correct action, the environment deterministically transitions to the next receptive field, S_2 , sequentially in order on the wafer map and provides a positive reward r . If the agent makes an incorrect decision, the environment instead presents a randomly chosen receptive field (for example, state S_7) from the next row of dies, with the associated penalty. An episode is defined as the complete path followed by the agent starting in state S_1 and terminating in state S_{term} , which is always the leftmost die in the bottom row of the wafer. In Fig. 3 for example, S_{76} is the terminating state. The metric of success in this environment is the cumulative reward achieved in an episode. The agent's objective is to learn an optimal policy that maximizes cumulative reward by correctly identifying all

dies (DUI, good, and bad) while ignoring out-of-wafer (white) regions. This formulation casts GDBN screening as an MDP, explicitly preserving sequential dependencies: at each step, the DQN agent transitions to the next state s' with reward r based on the previous state s and action a . Such an episodic setup ensures that wafer-level correlations and neighborhood context are captured. In Fig. 3, the RL agent starts at s_1 and proceeds along the yellow arrows until it reaches s_{76} , provided that each state is classified correctly. For clarity and aesthetics, the test parametric data are omitted from the figure. To further normalize the representation, the pixel values are scaled and the center die is masked to zero, encouraging the agent to exploit contextual features rather than trivially relying on the label of the central die.

B. Attention-based DQN Agent

This section describes in greater detail the multi-modal Attention-Based RL agent architecture, shown in Fig. 4. The processing of wafermap and ATE test data is carried out in two pipelines labeled MODE I and MODE II. The details of these two modality pipelines are discussed in the following.

a) Visual Modality Processing: Wafer maps exhibit distinct visual patterns that indicate potential defect locations throughout the wafer. These maps, composed of grayscale images, are treated as a vision modality due to their spatial characteristics. For each die under investigation (DUI), a local 3×3 neighborhood is extracted, forming a matrix $I_{\text{wf}} \in \mathbb{R}^{1 \times 3 \times 3}$, which encodes the test results of the target die and its eight immediate neighbors. To enhance the representational capacity of the input and capture die-aware contextual information, we apply a CNN-based self-attention mechanism inspired by Kapila et al. [31]. The input I_{wf} is linearly projected using learnable weights W_{q1} , W_{k1} , and $W_{v1} \in \mathbb{R}^{h \times c_{\text{Iwf}} \times w_{\text{Iwf}} \times h_{\text{Iwf}}}$, resulting in query, key, and value tensors Q_1 , K_1 , and V_1 , respectively:

$$Q_1 = I_{\text{wf}} \cdot W_{q1}^T, \quad K_1 = I_{\text{wf}} \cdot W_{k1}^T, \quad V_1 = I_{\text{wf}} \cdot W_{v1}^T \quad (2)$$

Here, h denotes the number of attention heads, c_{Iwf} is the number of channels in the input or depth of the input I_{wf} , while w_{Iwf} and h_{Iwf} represent the width and height of the input feature map, respectively. The self-attention output is

$$\text{Attention}(Q_1, K_1, V_1) = \text{Softmax}(Q_1 \cdot K_1^T) \cdot V_1. \quad (3)$$

This mechanism helps the model focus on relevant contextual information from neighboring dies, thereby improving its understanding of local defect patterns for defect inference in ambiguous GDBN regions. The resulting attention-enhanced feature is reshaped and passed through a convolutional layer. The convolution operation is defined as

$$\sum_{c_{\text{Iwf}}} \sum_m \sum_n Z_{c_{\text{Iwf}}}(c_x - m, c_y - n) K_{c_{\text{Iwf}}}(m, n), \quad (4)$$

where c_{Iwf} is the depth index for both the input feature map Z and the learnable kernel K , and (c_x, c_y) denote spatial coordinates. $K_{c_{\text{Iwf}}}(m, n)$ refers to the kernel value at position (m, n) at depth c_{Iwf} . Since the input wafer map is grayscale,

TABLE II
SYNTHESIZED NUMERICAL DATA INFORMATION.

Test Type	Unit	Pass Criteria	Fail Criteria
ROF	MHz	[32.57, 34.34]	[30, 32.57] \cup [34.34, 40]
IDDQ	μA	[0, 10]	[101, 130]
VminB	V	[0.9, 1.2]	[0, 0.6] \cup [1.31, 2]
VminS	V	[0.8, 1.0]	[0, 0.5] \cup [1.21, 2]
FMAX	GHz	[1, 5]	[0, 1]
PW	W	[0, 5]	[8, 15]
THRM	$^\circ\text{C}$	[0, 100]	[101, 200]

* ROF: Ring Oscillator Frequency; IDDQ: Quiescent Supply Current; VminB: Minimum Operating Voltage from Memory BIST Test; VminS: Minimum Operating Voltage of Scan at Speed Test; FMax: Maximum frequency of operation from logic At Speed Test; PW: Power Reading; THRM: Thermal Reading

the initial depth is $c_{\text{Iwf}} = 1$ and Z corresponds to the reshaped attention-based feature map. This convolutional layer extracts spatially significant features and produces an output of shape (batchSize, 24, 3, 3). The output is then passed through a batch normalization layer to normalize the activations, which reduces the internal covariate shift and improves training stability and generalization. A ReLU activation function is subsequently applied to introduce non-linearity. The activated output is then passed through a second convolutional layer, which applies the same operation described in (4). In this case, the input feature map Z has depth $c_{\text{Iwf}} = 24$, as it is the output produced by the previous convolutional layer. Again, the output is batch-normalized and passed through a ReLU activation function to complete the feature extraction process. The final output is denoted as Out_{wf} .

b) Numerical Modality Processing: During wafer-level chip probing, numerical test data is collected using automatic test equipment for various test types. We choose the seven tests listed in Table II. The self-attention output is calculated as

$$\text{Attention}(Q_2, K_2, V_2) = \text{Softmax}(Q_2 \cdot K_2^T) \cdot V_2. \quad (5)$$

The attention-enhanced feature is reshaped and sent through a convolutional layer.

c) Late Fusion and Suspiciousness Prediction: After obtaining the outputs from both modalities— Out_{wf} from the visual mode and Out_{num} from the numerical mode—the fused representation is obtained by concatenating the outputs as $F_{\text{fusion}} = \text{Out}_{\text{wf}} \oplus \text{Out}_{\text{num}}$, which is then passed through a fully connected layer with 256 neurons, followed by a ReLU activation function to introduce non-linearity. The output (Z_{fc}) is then fed into a final output layer with two neurons, and a softmax activation function (*Softmax*) is applied to calculate the probability of output actions. Later, the suspiciousness level $\text{SL} = \text{Softmax}(Z_{fc})$ is applied to classify as accept or reject. We choose late fusion over early fusion because it is well-suited for handling heterogeneous modalities and allows the model to learn independent, modality-specific representations before combining them.

C. Action Space and Reward Function

We cast wafer screening as a sequential decision process, where each die is evaluated within its local spatial context and

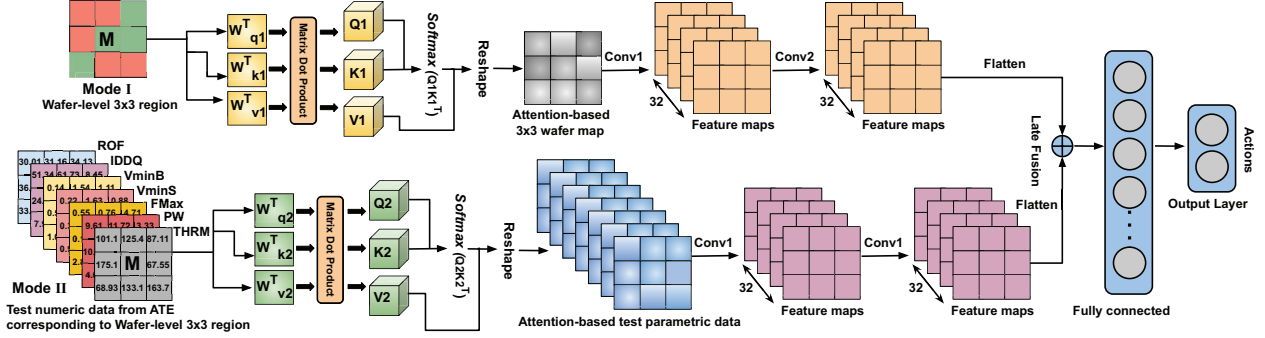


Fig. 4. The overall architecture of the attention-based RL agent.

the agent learns to optimize long-term yield–quality trade-offs. The framework follows an MDP defined by states, actions, rewards, and transitions. At each step, the agent selects one of $A = \{0, 1\}$ actions for a given state s :

- $a = 0$: accept the die (treat as non-defective),
- $a = 1$: reject the die (treat as defective).

This binary action space models the fundamental screening decision in the wafer test. Actions such as escalation or additional stress tests can also be included, but the present study focuses on the binary accept/reject setting.

The reward function is designed to reflect the asymmetric manufacturing costs of false rejects (yield loss) and false accepts (test escapes). Let the ground truth label of the die be $y_t \in \{0, 1\}$ (0 = good, 1 = defective) and the predicted action be a' . The reward r is defined as:

$$r = \begin{cases} +R_1, & \text{if } (a' = y_t = 1) \text{ (correct defect rejection)} \\ +R_0, & \text{if } (a' = y_t = 0) \text{ (correct acceptance)} \\ -R_1, & \text{if } (a' = 0, y_t = 1) \text{ (test escape)} \\ -R_0, & \text{if } (a' = 1, y_t = 0) \text{ (yield loss)} \end{cases}$$

where R_0 and R_1 are adaptive reward weights computed per wafer from the class imbalance. Specifically, the minority class receives larger magnitude rewards to counterbalance skewed distributions of good versus defective dies. This reward shaping ensures that the agent does not overfit to the majority class.

D. Replay Buffer

The replay buffer is crucial in stabilizing the training of RL agents by decoupling data collection from parameter updates. At each interaction step, the transition $(s, a, r, s', \text{done}, \text{label})$ is stored in the buffer, where s denotes the current state, a the selected action, r the reward received, s' the next state, done the termination flag, and label the ground-truth defect information. During training, the RL agent samples mini-batches of transitions uniformly (or with prioritization) from the buffer, rather than relying on strictly sequential data. This sampling strategy breaks the temporal correlations between consecutive states, reduces variance in updates, and improves data efficiency by reusing past experiences multiple times.

E. Smooth L1 Loss Optimization

To stabilize training, we use the smooth L1 loss (also known as the Huber loss) on the temporal-difference error. This loss function behaves quadratically for small errors and linearly for

large errors, thus combining the advantages of mean squared and mean absolute errors. Given a predicted Q-value $Q(s, a; \theta)$ and a Bellman target $y = r + \gamma \max_{a'} Q(s', a'; \theta)$, the TD error is defined as $\delta = y - Q(s, a; \theta)$. The loss is defined as

$$\text{SmoothL1}(\delta) = \begin{cases} \frac{1}{2}\delta^2, & \text{if } |\delta| < \beta, \\ \beta(|\delta| - \frac{1}{2}\beta), & \text{otherwise,} \end{cases} \quad (6)$$

where β is the transition threshold (default $\beta = 1.0$), δ_{clip} is a threshold parameter that controls the transition between the quadratic and linear regimes. Gradients of this loss are backpropagated through the DQN and the parameters θ are updated accordingly.

IV. EXPERIMENTAL RESULTS

We tested our framework on the WM-811K wafer dataset that contains 811,457 wafer map images collected from 46,293 real fabrication lots. Of these, 172,950 are manually labeled with eight defect patterns: Center, Donut, Edge-local, Edge-ringing, Local, Scratch, Random, and Near-full. To make a fair comparison with the SOTA methods, we use the same training and testing sample sizes and select a product with the highest data volume from the original WM-811k dataset [9], [10]. Specifically, we used 1,195,830 training samples and 297,780 test samples corresponding to wafers of size 25×27 .

To support multi-modal fusion, both spatial wafer bin maps and per-die numerical test data are needed from the WM-811K data set. Since no public data set currently includes numerical measurements along with wafer maps, we synthesize representative values for key test parameters. Synthesizing data is an accepted practice in machine learning to address challenges such as limited data availability, class imbalance, and privacy restrictions [32]–[37]. We synthesized the numeric data by sampling from Gaussian distributions whose means and variances are aligned with public domain wafer-level statistics. We selected seven representative test types commonly measured by test equipment, as listed in Table II. To simulate pass/fail outcomes, each synthetic test characteristic is stratified into “pass” or “fail” cohorts using the predefined thresholds in Table II. These values are then randomly assigned to the “good” or “bad” dies within the wafer maps. Although this stochastic mapping disrupts any true spatial-parametric correlation, it preserves the decision boundary between passing and failing profiles, which is the primary focus of this study.

TABLE III
COMPARISON OF TE COVERAGE, YIELD LOSS, GAIN, AND DPPM AT VARIOUS SUSPICIOUSNESS LEVELS RELATIVE TO SOTA METHODS.

Key Info	Reference	# TE	No GBDN	SL \geq 0.5	SL \geq 0.6	SL \geq 0.7	SL \geq 0.8	SL \geq 0.9
# Covered TE	[9]	86.3	0	55.0	52.2	49.2	45.6	40.1
	[9] ^[10]	86.3	0	27.7	22.9	18.3	13.0	7.0
	[Ours] ^{RL}	86.3	0	59.6	57.4	55.2	51.9	42.0
# Loss	[9]	86.3	0	18,769	15,417	12,505	9,630	6,549
	[9] ^[10]	86.3	0	9,668	5,594	3,088	1,271	297
	[Ours] ^{RL}	86.3	0	7,979	5,398	3,527	1,926	637
Yield Loss	[9]	86.3	0	6.3 %	5.2 %	4.2 %	3.2 %	2.2 %
	[9] ^[10]	86.3	0	3.2 %	1.9 %	1.0 %	0.4 %	0.1 %
	[Ours] ^{RL}	86.3	0	2.7 %	1.8 %	1.2 %	0.6 %	0.2 %
Gain	[9]	86.3	0	8,731	10,683	12,095	13,170	13,501
	[9] ^[10]	86.3	0	4,182	5,856	6,062	5,229	3,203
	[Ours] ^{RL}	86.3	0	21,842	23,325	24,078	24,014	20,343
DPPM	[9]	86.3	407.8	162.3	173.8	186.3	201.5	225.3
	[9] ^[10]	86.3	407.8	290.2	307.8	326.1	348.5	375.3
	[Ours] ^{RL}	86.3	407.8	129.4	138.4	147.9	162.6	208.6

* [9]^[10]: The authors of [9] used [10]’s methodology in their experiments; *RL*: Multi-modal RL approach developed in this paper.

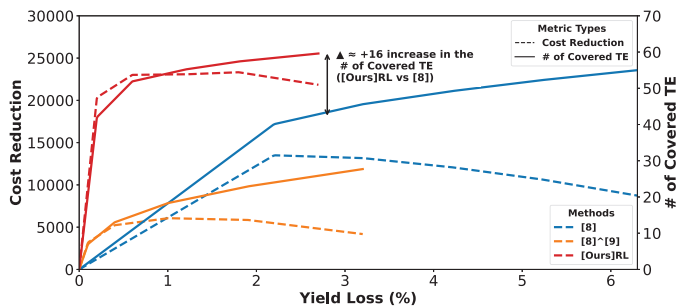


Fig. 5. Comparison of Cost and CTE with respect to SOTA

A. Discussion of Results

Table III presents a comparative analysis of our GBDN multi-modal RL framework with other methods in terms of various suspiciousness levels (SL). Performance metrics of interest include the number of covered test escapes (CTE), yield loss, gain (cost reduction), and outgoing DPPM, which collectively reflect both screening effectiveness and production impact.

Our RL approach substantially improves the number of covered test escapes across all SL thresholds compared to the SOTA approaches. For example, when SL is chosen to be ≥ 0.5 , CTE reaches 59.6, outperforming the two SOTA methods. This trend holds, with RL covering 42.0 TEs even at the strictest threshold (SL ≥ 0.9), demonstrating its robustness. Although the yield loss increases with lower thresholds (due to more aggressive rejection), RL maintains a favorable tradeoff to supervised methods. For example, RL limits the yield loss to 2.7% at SL ≥ 0.5 , which is 3.6 percentage points lower than the 6.3% loss achieved by Liu et al. [9]. The gain metric, which balances defect removal with yield sacrifice, highlights the superiority of RL. At SL ≥ 0.5 , RL achieves a gain of 21,560, which is more than double that of Liu et al. [9]. This advantage is maintained or widened at higher thresholds, validating the enhanced discriminative power afforded by multi-modal data fusion.

Figure 5 plots the relationship between yield loss, cost reduction (gain), and CTE for our method and SOTA baselines.

The left y-axis shows the cost reduction as a function of yield loss, while the right y-axis captures the number of covered test escapes over the same yield loss interval. When SL ≥ 0.5 , RL achieves a cost reduction of 21,842 units—more than 2.5 \times that of 8,731 achieved by Liu et al. [9]. As annotated within the plot, RL demonstrates improvement of 16 additional CTE’s over prior methods for the same wafer size 25×27 at a much better yield loss of 2.7%.

Finally, the RL approach delivers the lowest outgoing DPPM across all thresholds, achieving 129.4 at SL ≥ 0.5 , which is a reduction of 20% compared to the SOTA approaches for the same product size of 25×27 . It improves overall DPPM by 278 (from 407 when SL is set to 0, that is, no GBDN is applied) and achieves the highest DPPM reduction of 67.8%, outperforming the 58% reduction reported in Li et al. [21], 60.2% reported on a single product in Liu et al. [9], and 15.8% in Lu et al. [17]. Comparison of other metrics is not fair, as Li et al. [21] evaluate 11 distinct products, while our analysis is based on a single product, naturally leading to higher absolute TE counts in their case. Overall, the results demonstrate that the multi-modal attention-based RL agent provides a robust mechanism to improve test escape coverage while minimizing yield loss.

V. CONCLUSIONS

We have developed a RL-based framework for GBDN analysis which enables adaptive policies that balance yield and defect coverage. The experimental results demonstrate a much improved defect screening performance: 2.5X greater cost reduction and DPPM improvement of up to 20% in outgoing quality, while simultaneously reducing yield loss by 3.6 percentage points compared to SOTA approaches. For future work, this framework can be further extended to include delayed system-level test feedback, which identifies latent defects using a prioritized replay buffer.

REFERENCES

- [1] P.-C. Shih, A.-J. Su, K.-H. Tam, T.-C. Huang, K. Chuang, and J. Yeh, "Sow-x: A novel system-on-wafer technology for next generation ai server application," in *75th IEEE Electronic Components & Technology Conf. (ECTC)*, 2025, pp. 1–6.
- [2] S.-R. Chun, T.-H. Kuo, H.-Y. Tsai, C.-S. Liu, C.-T. Wang, J.-S. Hsieh, T.-S. Lin, T. Ku, and D. Yu, "Info sow (system-on-wafer) for high performance computing," in *ECTC*, 2020, pp. 1–6.
- [3] P. H. Hochschild, P. Turner, J. C. Mogul, R. Govindaraju, P. Ranganathan, D. E. Culler, and A. Vahdat, "Cores that don't count," in *Proceedings of the Workshop on Hot Topics in Operating Systems*, ser. HotOS '21. New York, NY, USA: Association for Computing Machinery, 2021, p. 9–16. [Online]. Available: <https://doi.org/10.1145/3458336.3465297>
- [4] M. E. Shaik *et al.*, "Predicting the silent data error prone devices using machine learning," in *2023 IEEE 41st VTS*. IEEE, 2023, pp. 1–4.
- [5] C. Constantinescu, I. Parulkar, R. Harper, and S. Michalak, "Silent data corruption — myth or reality?" in *IEEE Int'l Conf. Dependable Systems & Networks (DSN)*, 2008, pp. 108–109.
- [6] N. Srivarsan and K. Kempf, "Effective modeling of factory throughput times," in *Seventeenth IEEE/CPMT International Electronics Manufacturing Technology Symposium. 'Manufacturing Technologies - Present and Future'*, 1995, pp. 377–383.
- [7] A. Mishra, M. E. shaik *et al.*, "Wafercap: Open classification of wafer map patterns using deep capsule network," in *IEEE 42nd VTS*. IEEE, 2024, pp. 1–7.
- [8] A. K. Mishra *et al.*, "Wafer2spike: Spiking neural network for wafer map pattern classification," in *IEEE ITC*. IEEE, 2024, pp. 16–20.
- [9] C.-M. Liu *et al.*, "Enhancing good-die-in-bad-neighborhood methodology with wafer-level defect pattern information," in *IEEE ITC*, 2023, pp. 357–366.
- [10] C.-H. Yang *et al.*, "Identifying good-dice-in-bad-neighborhoods using artificial neural networks," in *IEEE 39th VTS*, 2021, pp. 1–7.
- [11] M. E. Shaik, A. Mishra, N. Kandasamy, and N. A. Toubia, "A multi-modal attention-based framework for good die in bad neighborhood methodology," in *2025 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, 2025, pp. 1–6.
- [12] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. MIT Press, 2018.
- [13] C. J. C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3–4, pp. 279–292, 1992.
- [14] V. Mnih, K. Kavukcuoglu, D. Silver, and *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
- [15] K. Community, "Wm-811k wafer map dataset," <https://www.kaggle.com/datasets/qingyi/wm811k-wafer-map>, 2018, accessed: 2025-09-13.
- [16] R. Miller *et al.*, "Unit level predicted yield: a method of identifying high defect density die at wafer sort," in *Proc. of ITC*, 2001, pp. 1118–1127.
- [17] C.-C. Lu *et al.*, "Transformer and its variants for identifying good dice in bad neighborhoods," in *IEEE VTS*, 2024, pp. 1–7.
- [18] J. Ngiam *et al.*, "Multimodal deep learning," in *ICML*, vol. 11, 2011, pp. 689–696.
- [19] C.-H. Chang and M.-C. Lee, "Reinforcement learning for semiconductor final test optimization," *IEEE Transactions on Semiconductor Manufacturing*, vol. 32, no. 3, pp. 333–342, 2019.
- [20] X. Zhang *et al.*, "Attention-based deep reinforcement learning for wafer defect pattern recognition," *IEEE Transactions on Semiconductor Manufacturing*, 2023.
- [21] S.-W. Li *et al.*, "Wafer-view defect-pattern-prominent gdbn method using metaformer variant," in *IEEE ITC*, 2024, pp. 76–80.
- [22] T. Baltrušaitis *et al.*, "Multimodal machine learning: A survey and taxonomy," *IEEE transactions on PAMI*, vol. 41, pp. 423–443, 2018.
- [23] A. Ghoreishee, A. Mishra, L. Zhou, J. Walsh, and N. Kandasamy, "New spiking architecture for multi-modal decision-making in autonomous vehicles," *arXiv preprint arXiv:2512.01882*, 2025.
- [24] A. N. Lingamoorthy, A. K. Mishra, S. Kumar, D. Gordon, N. Kandasamy, and A. Watson, "Drug overdose vital-signs evaluator using machine learning."
- [25] A. Mirhoseini, A. Goldie, M. Yazgan, J. W. J. Jiang, E. M. Songhori, S. Wang, Y. Lee, E. Johnson, O. Pathak, S. Bae, A. Nazi, J. Pak, A. Tong, K. Srinivasa, W. Hang, E. Tuncer, A. Babu, Q. V. Le, J. Laudon, R. Ho, R. Carpenter, and J. Dean, "Chip placement with deep reinforcement learning," *CoRR*, vol. abs/2004.10746, 2020. [Online]. Available: <https://arxiv.org/abs/2004.10746>
- [26] D. Zhao, S. Yuan, Y. Sun, S. Tu, and L. Xu, "Depth: Chip placement with deep reinforcement learning using a three-head policy network," in *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2023, pp. 1–2.
- [27] A. Dekhovich, O. Soloviev, and M. Verhaegen, "Sequential wafer map inspection via feedback loop with reinforcement learning," *Expert Systems with Applications*, vol. 275, p. 126996, 2025. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0957417425006189>
- [28] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.
- [29] A. Ghoreishee, A. Mishra, J. Walsh, A. Das, and N. Kandasamy, "Improving performance of spike-based deep q-learning using ternary neurons," *arXiv preprint arXiv:2506.03392*, 2025.
- [30] E. Barron and H. Ishii, "The bellman equation for minimizing the maximum cost," *Nonlinear Anal. Theory Methods Applic.*, vol. 13, no. 9, pp. 1067–1090, 1989.
- [31] N. Kapila *et al.*, "Cnnattention: Can cnns do better with attention?" *arXiv preprint arXiv:2412.11657*, 2024.
- [32] A. K. Mishra, A. K. Das, and N. Kandasamy, "Built-in functional testing of analog in-memory accelerators for deep neural networks," *Electronics*, vol. 11, no. 16, p. 2592, 2022.
- [33] J. Tremblay *et al.*, "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," in *Proceedings of the IEEE CVPR workshops*, 2018, pp. 969–977.
- [34] A. Mishra *et al.*, "Online performance monitoring of neuromorphic computing systems," in *IEEE ETS*. IEEE, 2023, pp. 1–4.
- [35] A. Mishra, A. Das, and N. Kandasamy, "Model-based approach towards correctness checking of neuromorphic computing systems," in *2024 IEEE 29th Pacific Rim International Symposium on Dependable Computing (PRDC)*. IEEE, 2024, pp. 11–21.
- [36] S. Kumar, A. Mishra, A. Das, and N. Kandasamy, "Hierarchical model-based approach for concurrent testing of neuromorphic architecture," in *2025 55th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2025, pp. 511–523.
- [37] A. K. Mishra, "Verification of neuromorphic computing hardware using machine learning," Ph.D. dissertation, Drexel University, 2025.