PoLM: Point Cloud and Large Pre-trained Model Catch Mixed-type Wafer Defect Pattern Recognition

Hongquan He[1] Guowen Kuang[2] Qi Sun[3] Hao Geng[1,4§]
[1]ShanghaiTech University [2]Shenzhen Polytechnic [3]Zhejiang University
[4]Shanghai Engineering Research Center of Energy Efficient and Custom AI IC

***Abstract*—As the technology node scales down to 5nm/3nm, the consequent difficulty has been widely lamented. The defects on the surface of wafers are much more prone to emerge during manufacturing than ever. What's worse, various single-type defect patterns may be coupled on a wafer and thus shape a mixed-type pattern. To improve yield during the design cycle, mixed-type wafer defect pattern recognition is required to perform to identify the failure mechanisms. Based on these issues, we revisit failure dies on wafer maps by treating them as point sets in two-dimensional space and propose a two-stage classification framework, PoLM. The challenge of noise reduction is considerably improved by first using an adaptive alpha-shapes algorithm to extract intricate geometric features of mixed-type patterns. Unlike sophisticated frameworks based on CNNs or Transformers, PoLM only completes classification within a point cloud cluster for aggregating and dispatching features. Furthermore, recognizing the remarkable success of large pre-trained foundation models (*e.g.*, OpenAI's GPT-n series) in various visual tasks, this paper also introduces a training paradigm leveraging these pre-trained models and fine-tuning to improve the final recognition. Experiments demonstrate that our proposed framework significantly surpasses the state-of-the-art methodologies in classifying mixed-type wafer defect patterns.**

I. Introduction

In modern semiconductor manufacturing, dies on the wafer are scanned and tested by the inspection probe to see if their fundamental functionality is error-free. Each die is then assigned a binary value indicating whether it passes or fails the test. The values on a wafer will be presented as binary pixels in 2 dimensions, known as wafer bin map (WBM). In most cases, failure dies on a wafer map are prone to converge into a particular distribution pattern type. These defect patterns, such as randomly distributed, linear, curved, and circular patterns, are associated with certain manufacturing processes. Recognizing these patterns at an early fabrication stage makes it possible to locate and reason issues, thereby helping to improve chip yield.

There exist a vast number of prior works to recognize single-type wafer defect patterns. In some works, WBMs are usually considered as single-channel images to the visual classification models and their spatial properties are used to identify the wafer defects [1]. Recently, deep learning-based methods [2] are widely embraced for defect classification for their outstanding image interpretability and noise robustness.

However, with the increased number of integrated circuits per unit area, various single-type defect patterns are prone to be coupled on a wafer and thus shape a mixed-type pattern [3].

§ Corresponding author

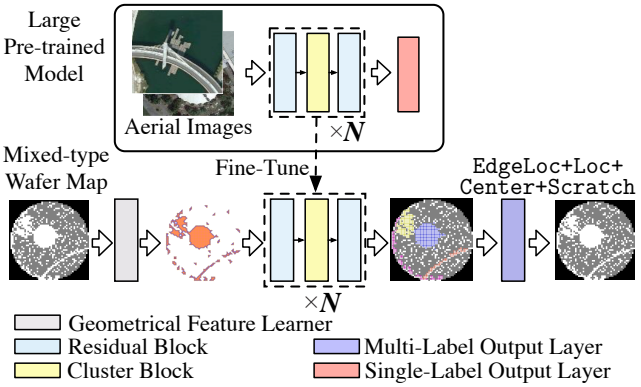


Fig. 1 The proposed mixed-type wafer defect pattern recognition framework. The geometrical feature learner transforms WBM into polygon-based features, and then residual blocks and cluster blocks are well designed into subsequent deep learning framework for feature aggregation and dispatching. With each failure die assigned to a separate cluster at the die-level, parameters from a large pre-trained model are partially incorporated into PoLM for fine-tuning.

Different from single patterns on WBMs, mixed-type defect patterns are more difficult to recognize. Despite the importance of recognizing mixed-type wafer defect patterns, few works [4], [5] exploiting CNN and transformer architectures have been proposed to address the dilemma.

It could be viewed that several issues exist in previous works. Firstly, existing feature extraction methods are sensitive to noise, according to results from [1]. Secondly, convolutional neural networks (CNNs) disregard the impression of the WBM's overall perspective, leading to poor recognition performance on mixed-type defect patterns globally. Besides, although vision transformers (ViTs) [6] are introduced in [5], each input/patch is required to generate through a CNN-based block, resulting much higher cost for training. Finally, high-performing classifiers for defect pattern recognition [2] may sometimes live in the shadow of state-of-the-art works in image vision, sparking our pursuit of a simple yet powerful model.

In this paper, to overcome the aforementioned issues and to address the mixed-type wafer defect pattern recognition better, we propose a novel framework PoLM based on the new generation of artificial intelligence (AI) technologies. Our contributions are summarized as follows:

- The challenge of noise reduction for mixed-type patterns

979-8-3503-4859-0/DATE24/© 2024 EDAA

is considerably improved by first using an adaptive alpha-shapes learner to extract intricate geometric features.
- The binary pixels in WBMs are treated as point sets in 2-dimension space which are then handled by proposed cluster-based model in the point cloud.
- We leverage a large pre-trained model as the prototype of our defect recognition model, with dedicated fine-tuning to improve the final recognition quality.

The rest of the paper is organized as follows. Section II introduces some prior knowledge of mixed-type wafer failure pattern recognition and then provides a problem formulation. Section III-A sketches our geometrical feature learner via alpha-shapes algorithm. Section III-B describes the deep learning based flow with cluster blocks and resblocks in proposed classification framework. Section III-C depicts our training paradigm inspired by large pre-trained foundation models, while Section IV presents the experimental results followed by conclusion in Section V.

II. Preliminaries

In this work, we take into account `MixedWM38` industrial wafer bin map (WBM) dataset [4] in Fig. 2, covering 38 single and mixed-type defect patterns. 9 single-type patterns are labeled in one-hot encoding and mixed-type patterns are coupled by 6 of them (with the exception of `Normal`, `Random`, and `NearFull`). There are 13, 12, and 4, respectively, of the 2/3/4-mixed-type pattern. Some WBMs in `MixedWM38` were generated using GANs to address the issue of unbalanced sampling of various failure patterns acquired in practice.

The majority of arts treat WBMs as single-channel images with three possible pixel levels for each die: 0, 127, and 255, which represent different physical meanings, including not being a part of the wafer, die locations with a pass label, and die locations with a failure label. The gathering of failure dies at multiple sites to pair certain patterns is the essence of defect patterns. Within each defect pattern, the pixel points do not exhibit any color differences apart from variations in position and quantity. Therefore, we may re-encode WBMs using the structure of point cloud, where pixels with level 0 and 127 are considered as the background while pixels with level 255 are failure dies shown in Definition 1.

A thorough strategy for the classification of mixed-type patterns (*i.e.* Problem 1) uses recall, accuracy, and F_1 score as assessment measures. These metrics convey substantial insight about how well the model performs in terms of recognizing various defect patterns in WBMs.

Definition 1 (WBM as a set of points)**.** *Given a WBM* $M \in \mathbb{R}^{1\times w\times h}$*, the locations of each pixel* m_{ij} *can be normalized and encoded as* $\text{Position}\,(m_{ij}) = \left[\frac{i}{w}-0.5, \frac{j}{h}-0.5\right]^T$*, s.t.* $\frac{i}{w}, \frac{j}{h} \in [0,1]$*. Taking into count pixel levels, image-based WBMs can be converted into a series of* 3*-channel points* $\mathbb{M} = \mathbb{W} \cup \mathbb{W}' \in \mathbb{R}^{3\times n}$ *within 1 color and 2 locations elements, where* $\mathbb{W}$ *are failure dies within level* 255 *while* $\mathbb{W}'$ *are background.*

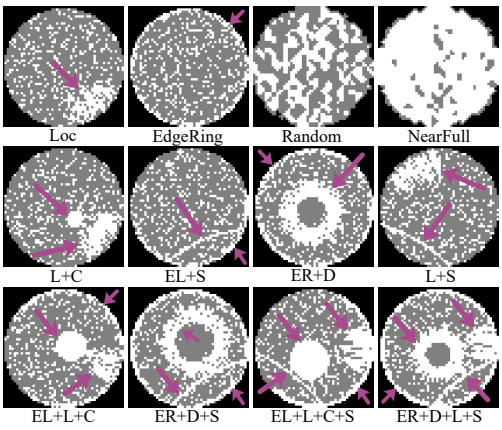


Fig. 2 Single and mixed-type of wafer pattens in the benchmark. The first row shows 4 single-type patterns: `Loc`, `EdgeRing`, `Random`, and `NearFull`, where `Random` and `NearFull` are specific patterns that will never be coupled. The second row shows mixed-type patterns formed by the overlap of 2 single-type patterns: `L+C`, `EL+S`, `ER+D`, and `L+S`, where `L`, `C`, `D`, `S`, `ER`, `EL` are the pattern that may be coupled. The first two in the third row are in 3 single-type patterns, while the last two are coupled with 4 single-type patterns.

Problem 1 (Mixed-type Wafer Defect Pattern Classification)**.** *Given a set of WBMs* $\mathcal{M} = \{\mathbb{M}_1, \mathbb{M}_2, \cdots, \mathbb{M}_{n_1}\}$ *and a labeling space* $\mathcal{Y} = \{y_1, y_2, \cdots, y_{n_2}\}$*, our work is to build a deep learning model* $\mathcal{H} : \mathcal{M} \to \mathcal{Y}$ *from training set* $\{\mathcal{D} = (\mathbb{M}_i, Y_i), i \in [0, n_1 - 1]\}$*, where* $\mathbb{M}_i \in \mathcal{M}$ *is a WBM and* $Y_i \subset \mathcal{Y}$ *is a subset of labeling sapce. For a unlabeled WBM* $\mathbb{M} \in \mathcal{M}$*, multi-label classifier* $\mathcal{H}(*)$ *predicts* $\mathcal{H}(\mathbb{M} \to \mathcal{Y})$ *as recognition results.*

III. The Developed Detection Flow

Classification of mixed-type failure patterns is crucial to boosting industry yield. Since various failure pattern types are essentially diverse geometric shapes aggregated by failure dies on the wafer map, we explain the geometric characteristics of mixed-type failure patterns from a point cloud perspective and identify them through a clustering framework. `PoLM` can also recognize mixed-type patterns that have not been included in the training set as the geometric characteristics solely depend on single-type patterns and are not sensitive to which combination of them. We also introduce a pre-trained paradigm to fine-tune our cluster-based flow as it has been achieving great success in vision tasks.

A. The geometrical feature learner

Failure dies mostly come in two varieties on the wafer map. One is a part of a single-type pattern, whereas the other is a distinct fault site. The alpha-shapes algorithm [7] is implemented to acquire the former's geometric attributes and lessen the latter's noise impact on defect pattern detection.

We first constitute a few concepts before delivering our proposed geometrical feature learner in Fig. 1. For failure dies $\mathbb{W} = \{W_1, W_2, ..., W_n\}$ on a wafer map, the convex hull of

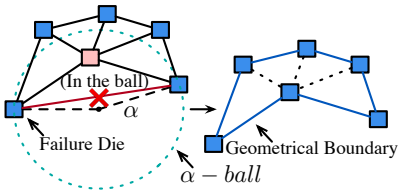
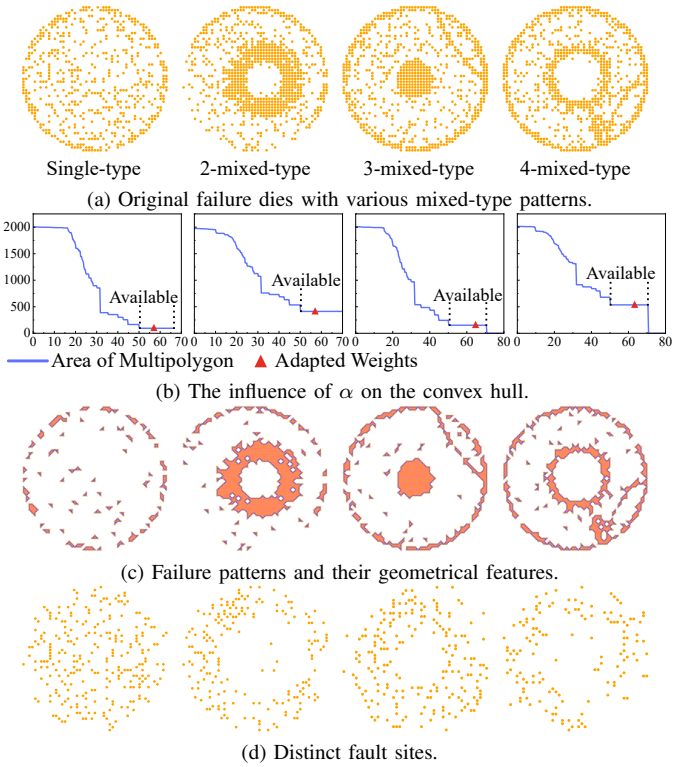


Fig. 3 Illustration of geometrical feature learner.

$\mathbb{W}$ are required to be triangulated to generate the set of the triangle structure $\boldsymbol{T} = \{\triangle T_1, \triangle T_2, ..., \triangle T_m\}$ that satisfies: all endpoints in $\boldsymbol{T}$ exactly constitute $\mathbb{W}$, any 2 sides in $\boldsymbol{T}$ do not intersect, and $\boldsymbol{T}$ reveals the convex hull of $\mathbb{W}$. It depicts a `delaunay_triangulation` [7] that is fully formed in the left part of Fig. 3. The vertices of the convex hull are made up of the blue failure dies, and this stable triangular arrangement has satisfied the empty circle property [8].

Generally, The polygons $\partial \boldsymbol{T}$ built after reconstructing the point cloud by `delaunay_triangulation` are typically imprecise and unable to capture the geometrical traits. In order to roughly estimate the concave hull area of failure patterns, we further consider the `alpha_shapes` technique and tweak the alpha threshold (`adapt_alpha`) in Algorithm 1.

A Delaunary Triangulation (line 2 in `alpha_shapes`) is first built using failure dies $\mathbb{W}$ in order to simplify complexity, and the judgment is then initiated from its outer border as Fig. 3. It indicates that a triangle $\triangle T$ is removed if one of its sides is longer than 2α. Each side of convex hull $\boldsymbol{T}$ is assessed simultaneously. The triangle is also eliminated if there are any failure dies in a α-ball with a radius of α that passes through a particular side (line 4). Locate the boundary of the remaining triangulations, which is the failure pattern's geometrical characteristic (line 7), to finish. With regard to failure dies dispersed unevenly on the wafer, constructing alpha-shapes offer a reliable geometrical feature learning technique.



(a) Original failure dies with various mixed-type patterns.

(b) The influence of α on the convex hull.

(c) Failure patterns and their geometrical features.

(d) Distinct fault sites.

Fig. 4 Illustration of mixed-type failure patterns and their geometrical features, which is not sensitive to pattern types.

Algorithm 1 Geometrical Feature Extraction of Failure Dies

Input: A set of failure dies $\mathbb{W}$; Radius of $\alpha - ball$ α.

1: **function** `alpha_shapes`($\mathbb{W}$, α)
2: $\boldsymbol{T} \leftarrow$ `delaunay_trianglation`($\mathbb{W}$);
3: **for** $\triangle T \in \boldsymbol{T}$ **do**
4: **if** any side $\in \triangle T > 2\alpha$ **or** any point inside $\alpha - ball$ with radius α **then** $\boldsymbol{T} \leftarrow \boldsymbol{T} - \triangle T$; ▷ Fig. 3
5: **end if**
6: **end for**
7: **return** $\boldsymbol{I} \leftarrow \partial \boldsymbol{T}$;

Output: Polygon-based wafer image $\boldsymbol{I}$.

Input: A set of failure dies $\mathbb{W}$; Increased and decay factor λ, β; Decay step k; Number of max-iterations N.

1: **function** `adapt_alpha`($\mathbb{W}$, λ, β, k, N)
2: $\alpha \leftarrow$ radius of a wafer, $j \leftarrow 0$;
3: **while** $t < N$ **do**
4: $\alpha \leftarrow (1-\lambda)\alpha$, $j \leftarrow j+1$;
5: Update S for $S' \leftarrow$ area of `alpha_shapes`($\mathbb{W}, \alpha$);
6: **if** j mod $k = 0$ **then** $\lambda \leftarrow \beta\lambda$, $j \leftarrow 0$;
7: **end if**
8: **if** $|S' - S|$ is convergent **then** $t \leftarrow t+1$;
9: **end if**
10: **end while**
11: **return** $\boldsymbol{F} \leftarrow$ `alpha_shapes`($\mathbb{W}, \alpha$);

Output: Geometrical Feature $\boldsymbol{F}$.

Then, when it comes to mixed-type patterns, the amount and type of patterns have an impact on the precise distribution of failure dies. While most ML-models have tricks to dynamically adjust hyper-parameters, `adapt_alpha` in Algorithm 1 can be regraded as a similar method that returns the polygon-based feature with the best performance on various dies distribution. Indicator j will count up while the radius of $alpha$-ball steps down through iterations (line 4), and in each iteration, the area of polygon-based wafer image S is calculated and updated for alpha-shapes (line 5). When j is an integer multiple of k, λ is reduced to $\beta\lambda$ and the radius α decays every k iterations (line 6-9). At the end of `adapt_alpha` with weights in Fig. 4(b), an adapted geometrical boundary $\boldsymbol{F}$ of failure patterns will be return (line 11). We visualize 4 mixed-type patterns coupled with various single-type patterns in Fig. 4, where failure dies in Fig. 4(a) are separated into polygons in Fig. 4(c) and distinct fault sites in Fig. 4(d) with our geometrical feature learner. This approach does not depend on pattern labels, fully overcomes the variety of numbers and their combinations, and provides a strong geometric extractor on die level.

B. Cluster-based learner

With no blatant color changes, polygon-based features of failure dies depart from natural images (*e.g.* ImageNet dataset), and failure patterns in categorization should look at polygonal shape changes globally rather than locally. In addition, failure patterns gathered via the expertise of IC engineers are also pricey and challenging to scale up. This motivates us to develop a deep learning flow based on global clustering [9] to recognize patterns on die level rather than a CNN or Transformer-based paradigm.

To aid in comprehension, we sketch the diagram of the cluster-based learner in Fig. 5. Polygon-based feature $\boldsymbol{F}$ is embedded into a set of randomly feature points with 3 channels (colors and positions). Then, N reapting stages with residual block and cluster block will extract and convey global features of existing patterns. Equation (1) illustrates the process of mixed-type pattern classification in the framwork.

$$f_i = \Phi_{pos}\left(\mathcal{C}\left(\Phi_{pre}\left(s_{ij}\right) \mid j = 1, 2, \ldots, \mathrm{K}\right)\right), \tag{1}$$

where s_{ij} is one embedded feature in cluster j and f_i is i^{th} communicated features of failure polygons, $\mathcal{C}$ represents cluster block while Φ_{pre} and Φ_{pos} are residual block at the beginning and ending of each cluster block shown in Fig. 5. Predicted labels return when deep features communicate through fully connected layers.

ResBlock shown at left in Fig. 5, as a relatively mature module in point cloud classification tasks [10], relies on MLPs and residual connections for feature fusion. Since multilayer perceptron (MLP) layers are used to replace CNNs, it is inherently free of spatial restrictions, which fits well with the properties of random and dispersed failure dies. Each of resblocks is composed of conditional batch normalization (BN), ReLU activation function, and 2 MLP layers to transmit features among s_{ij} constructed as,

$$\text{MLP}: Linear \rightarrow (Linear + ReLU) \times M \cdots \rightarrow Linear, \tag{2}$$

where $\times M$ means there are M hidden blocks for $Linear \rightarrow ReLU$ in embedding space while input and output of resblock are residual connected. A ReLU activation constructed as,

$$\text{LeakyRelu}(x) = \left\{ \begin{array}{ll} x, & x > 0 \\ \alpha \times x, & x \leq 0 \end{array} \right\}, \tag{3}$$

is transferring the fused feature where α is set as 0.01.

The cluster block in the right of Fig. 5 works when there is feature interaction between various failure dies. This block distributes and collects features within each cluster using points as input. The process of obtaining context-information includes three steps: feature clustering, feature aggregation, and feature dispatch. It begins by uniformly initializing c cluster centers within input feature points $\mathbb{P} \in \mathbb{R}^{n \times d}$ in the spatial domain d. Cosine similarity matrix $\boldsymbol{S} \in \mathbb{R}^{c \times n}$ between centers and other points is then calculated as a criterion for determining the proximity among points. Each element s in $\boldsymbol{S}$ is shown in Equation (4),

$$s = \frac{\sum_{i=1}^{n}\left(\boldsymbol{r}_i \times \boldsymbol{t}_i\right)}{\sqrt{\sum_{j=1}^{n}\left(\boldsymbol{r}_j\right)^2} \times \sqrt{\sum_{j=1}^{n}\left(\boldsymbol{t}_j\right)^2}}, \tag{4}$$

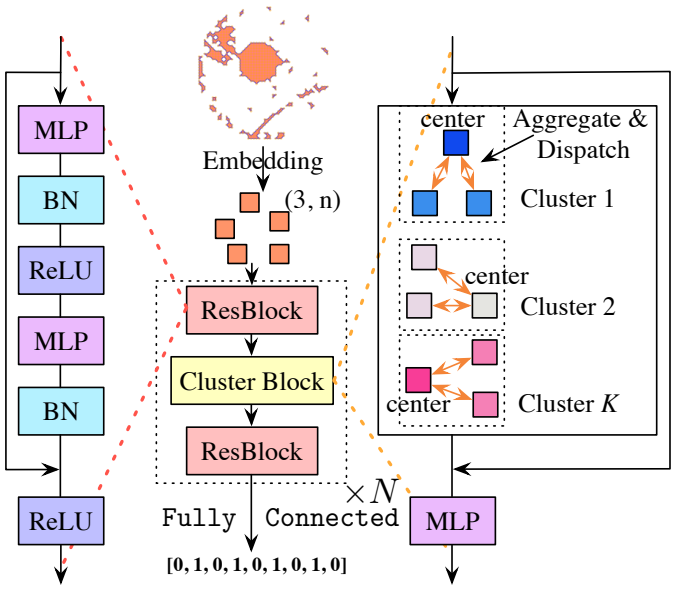


Fig. 5 The visualization of cluster-based learner.

where $\boldsymbol{r}$ and $\boldsymbol{t}$ are two feature points in dimension d.

Generally, KNN (K-Nearest Neighbors) is to assemble the feature points $\mathbb{P}$ and forms c clusters ultimately, where each of them is assigned to the cluster whose center is most similar to it. Based on the similarity $\boldsymbol{S}$ to centers, points within each cluster are dynamically aggregated to the cluster center according to Equation (5),

$$\begin{aligned} \boldsymbol{g} &= \frac{1}{C}\left(\boldsymbol{v_c} + \sum_{i=1}^{m} sig(\alpha s_i + \beta) \times \boldsymbol{v_i}\right), \\ \text{s.t. } C &= 1 + \sum_{i=1}^{m} sig\left(\alpha s_i + \beta\right) \end{aligned} \tag{5}$$

where $\boldsymbol{v_i}$ is the i^{th} point in Cluster v, and s_i represents cosine similarity in Equation (4) between $\boldsymbol{v_i}$ and the center $\boldsymbol{v_c}$. The function $sig(*)$ refers to the Sigmoid activation, which is used to limit the similarity values within the range of 0 to 1. λ and β are learnable parameters for scaling and biasing the similarity values. This process involves adding the information from each $\boldsymbol{v_i}$ to the center $\boldsymbol{v_c}$, and then taking the average.

After features are aggregated to the center $\boldsymbol{v_c}$, it is essential to facilitate feature communication and learning from other failure points. For the i^{th} point p_i in cluster v, distributing $\boldsymbol{g}$ is equivalent to conducting implicit relationship learning for every point in the cluster as,

$$p_i' = p_i + FC(sig(\lambda s_i + \beta) \times \boldsymbol{g}), \tag{6}$$

where $\boldsymbol{g}$ represents the globally aggregated feature from center $\boldsymbol{v_i}$ in Fig. 5 and $FC(*)$ refers to a fully connected layer. To distribute this feature to p_i (points marked with light color in Fig. 5), it is multiplied with the similarity s_i between $\boldsymbol{g}$ and p_i to yield features in $\boldsymbol{g}$ allocated to p_i. It serves as the inverse process of feature aggregation, where features assigned from $\boldsymbol{g}$ to p_i is added to the original information contained in p_i, resulting in the distributed feature point p_i'.

TABLE I Benchmark Statistics

Categories	Training Set		Testing Set	
	#	Percent (%)	#	Percent (%)
`Random` (C7)	606	2.277	260	2.277
`NearFull` (C9)	104	0.391	45	0.391
`C+EL+S` (C24)	1400	5.261	600	5.261
`Others` (×35)	700	2.630	300	2.630
`Total` (C1-C38)	103767	100	69183	100

Ultimately, the cluster block activates communication and interaction between failure regions via aggregating and dispatching features, allowing them to be conveyed further in a MLP layer at last.

C. Large pre-trained learner

LLMs (*e.g.* OpenAI's GPT-n) and vision models (*e.g.* Meta's Segment Anything [11]) both aim to converge all AI tasks. Those works reveal pre-trained models (PTMs) in larger datasets can overcome obstacles caused by the scarcity of relevant benchmarks. As a pre-trained source model is only fine-tuned during training with few data, ensuring similarity between the source and target datasets is essential for the effective implementation of the PTM.

Coincidentally, failure patterns consist of a variety of irregular polygons and have no variation in color when some aerial scene datasets like `AID` [12] also contain similar characteristics. The shape of category `baseball field` and `bridge` in `AID` shown in Fig. 1 is like a ring and a scratch (refer to `Dount` and `Scratch` in failure patterns). Generally, we pre-train a model by the `AID` benchmark to offer a good starting point for our specified mission of pattern recognition, for the sake of its robust weights and biases.

IV. Experimental Results

A. Experimental Settings and Benchmark Statistics

Our framework is implemented in `Pytorch` library, and it is tested on a platform with the Nvidia RTX 3060 Graphics card and an Intel Core $i9-10900k$ processor. The industry benchmark mentioned in Section II, `MixedWM38`, is used to verify performances of our pattern recognition flow, `PoLM`. There are 38015 wafer images from 46293 lots covering 38 categories of patterns annotated by fab experts. We summarize the statistics of dataset in TABLE I, where Cn is one pattern category, and sample numbers of 35 patterns are the same shown in `Others`, except for C7, C9, C24. The whole labeled wafer maps is split into $0.7:0.3$ for training and testing.

Regarding the training details, all of our models are trained using `AdamW` across 500 epochs with a momentum of 0.9 and a weight decay of 0.05. The learning rate is set to 0.001 and is changed using a cosine schedular.

B. Comparisons Against SOTA Works

We compare our proposed classifier against one deformable CNN-Based work [4] (refers to "TSM'20"), one transformer-based art [5] (refers to "TSM'22"), and the cutting-edge Segment Anything Model [11] released by Meta (refers to "Meta'23"). Following the evaluation metrics in previous arts, we utilize recall, precision rate and F_1 as evaluation metrics to fully examine the true states of the models' performance.

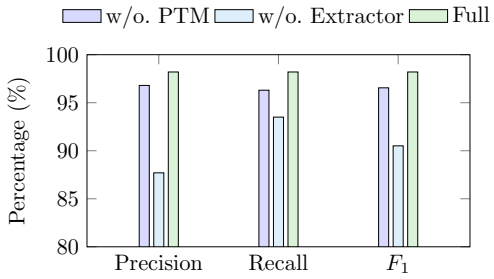


Fig. 6 Ablation study results. The light bars illustrate Precision, Recall, and F_1 Score with the framework removed PTM, geometrical feature learner, and whole `PoLM` respectively.

TABLE II records the results of three methods and their corresponding macro-averaged values (i.e. arithmetic means). We can see our method macro-averagely outperforms TSM'20 with both 5.3% improvement on precision and recall and 5.4% rise on F_1 score. It surpasses TSM'22 with a macro-averaged precision, recall and F_1 score of 3.6%, 3.3% and 3.5%, respectively. It is worth noticing that our method behaves much better than Meta'23 on all evaluation metrics averagely while we just fine-tune the image encoder block in SAM to extract WBMs' features and do not consider the priori information from prompt encoder. This reveals that there is still a long way to go for general large pre-trained vision models for a specific EDA task like mixed-type pattern recognition. A specially designed detector framework is needed for the specific tasks that require a lot of domain knowledge. Overall, our approach achieves 98.2% macro-averaged recall, precision rate and F_1 score (shown in TABLE II), which exceeds more than those of its counterparts. Furthermore, our inference time on the 3060 Graphics is an average of 13.51ms, outperforming most works [4], [5], thanks to the simple flow of cluster-based learner.

C. Abalation Studies

We also look into how various setups impact the performance of the macro-averaged classification. Fig. 6 outlines the contributions of large pre-trained model (PTM) and geometrical feature learner paradigm to our flow. "w/o. PTM" refers to the flow trained without PTM to fine-tune our parameters, and "w/o. Extractor" stands for the flow trained without eometrical feature learner, while "Full" is our wafer failure pattern classifier `PoLM` with entire techniques. The histogram suggests that with lack of PTM, all macro-averaged metrics is slightly affected. This may be due to the large variability of natural RGB-image datasets `AID` [12] and WBMs. As demonstrated by the bar graph, recall is solely marginally influenced by the absence of the geometrical feature learner technique, but there is a significant decline in both the precision rate and the F_1 score. This explanation highlights the significant challenges posed by scattered failure dies on the wafer in pattern recognition.

TABLE II Comparison with state of the arts

Pattern	CNN-Based (TSM'20 [4])			Transformer-Based (TSM'22 [5])			Image Encoder in SAM (Meta'23 [11])			PoLM (Ours')		
	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score	Precision	Recall	F1 Score
C1	0.929	0.917	0.923	0.974	**1.000**	0.987	0.974	0.993	0.983	**0.993**	**1.000**	**0.997**
C2	0.870	0.957	0.911	0.936	0.973	0.954	0.905	0.957	0.930	**0.974**	**1.000**	**0.987**
C3	0.965	0.923	0.944	0.967	0.973	0.970	0.976	0.947	0.961	**0.984**	**0.997**	**0.990**
C4	0.967	0.890	0.927	0.976	0.937	0.956	0.983	0.937	0.959	**0.990**	**0.997**	**0.993**
C5	0.923	0.960	0.941	0.938	0.963	0.951	0.930	0.970	0.949	**0.987**	**0.990**	**0.988**
C6	0.904	0.970	0.936	0.961	0.977	0.969	0.970	0.957	0.963	**0.983**	**0.993**	**0.988**
C7	0.911	0.950	0.930	0.992	0.950	0.971	0.961	0.950	0.956	**0.996**	**0.996**	**0.996**
C8	0.863	0.923	0.892	0.954	0.970	0.962	0.948	0.970	0.959	**0.997**	**0.997**	**0.997**
C9	0.750	0.867	0.804	0.759	0.911	0.828	0.729	0.778	0.753	**0.977**	**0.933**	**0.955**
C10	0.931	0.947	0.939	0.954	0.960	0.957	0.950	0.947	0.948	**0.987**	**0.987**	**0.987**
C11	0.913	0.980	0.945	0.951	0.977	0.964	0.918	0.977	0.947	**0.980**	**0.980**	**0.980**
C12	0.942	0.927	0.934	0.960	0.963	0.962	0.948	0.963	0.955	**0.987**	**0.983**	**0.985**
C13	0.907	0.943	0.925	0.930	0.973	0.951	0.910	0.973	0.940	**0.980**	**0.990**	**0.985**
C14	0.933	0.923	0.928	0.944	0.957	0.950	0.938	0.957	0.947	**0.990**	**0.980**	**0.985**
C15	0.896	0.950	0.922	0.917	0.960	0.938	0.908	0.953	0.930	**0.974**	**0.983**	**0.978**
C16	0.915	0.933	0.924	0.935	0.953	0.944	0.934	0.947	0.940	**0.977**	**0.980**	**0.978**
C17	0.960	0.953	0.957	0.956	0.950	0.953	0.963	0.960	0.962	**0.993**	**0.990**	**0.992**
C18	0.870	0.960	0.913	0.912	0.970	0.940	0.877	0.950	0.912	**0.967**	**0.983**	**0.975**
C19	0.900	0.933	0.917	0.972	0.933	0.952	0.946	0.933	0.940	**0.997**	**0.980**	**0.988**
C20	0.930	0.907	0.918	0.927	0.967	0.946	0.922	0.930	0.926	**0.964**	**0.993**	**0.979**
C21	0.968	0.907	0.936	0.960	0.960	0.960	0.959	0.930	0.944	**0.983**	**0.987**	**0.985**
C22	0.962	0.917	0.939	0.963	0.963	0.963	0.964	0.970	0.967	**0.984**	**0.997**	**0.990**
C23	0.976	0.947	0.961	0.979	0.953	0.966	0.976	0.943	0.959	**0.997**	**0.970**	**0.983**
C24	0.972	0.911	0.941	0.988	0.977	0.982	0.986	0.965	0.976	**0.997**	**0.990**	**0.993**
C25	0.877	0.930	0.903	0.873	0.940	0.905	0.869	0.930	0.899	**0.951**	**0.980**	**0.966**
C26	0.969	0.930	0.956	0.973	0.977	0.975	0.969	0.943	0.956	**0.983**	**0.993**	**0.988**
C27	0.969	0.947	0.958	0.983	0.937	0.959	0.983	0.940	0.961	**0.986**	**0.970**	**0.978**
C28	0.943	0.880	0.910	0.937	0.937	0.937	0.927	0.927	0.927	**0.990**	**0.977**	**0.983**
C29	0.963	0.960	0.962	0.951	0.903	0.926	0.944	0.903	0.923	**0.973**	**0.973**	**0.973**
C30	0.929	0.960	0.944	0.970	0.957	0.963	0.962	0.933	0.948	**0.993**	**0.987**	**0.990**
C31	0.911	0.923	0.917	0.884	0.940	0.911	0.877	0.923	0.899	**0.967**	**0.970**	**0.968**
C32	0.964	0.887	0.924	0.968	0.917	0.942	0.962	0.917	0.939	**0.980**	**0.973**	**0.977**
C33	0.961	0.903	0.931	0.978	0.910	0.943	0.961	0.893	0.926	**0.997**	**0.963**	**0.980**
C34	0.955	0.930	0.943	0.935	0.917	0.926	0.933	0.923	0.928	**0.980**	**0.960**	**0.970**
C35	0.954	**0.967**	0.960	0.959	0.930	0.944	0.941	0.910	0.925	**0.980**	**0.967**	**0.973**
C36	0.962	0.853	0.904	0.932	0.863	0.896	0.926	0.870	0.897	**0.963**	**0.950**	**0.956**
C37	0.948	0.910	0.929	0.965	0.910	0.937	0.947	0.887	0.916	**0.980**	**0.970**	**0.975**
C38	0.922	0.940	0.931	0.944	0.953	0.949	0.926	0.953	0.939	**0.971**	**0.990**	**0.980**
Avg.	0.929	0.929	0.928	0.946	0.949	0.947	0.937	0.937	0.937	**0.982**	**0.982**	**0.982**
Ratio	0.945	0.947	0.945	0.963	0.967	0.965	0.954	0.955	0.954	**1.000**	**1.000**	**1.000**

V. Conclusion

In this paper, we have proposed PoLM, a mixed-type wafer defect pattern recognition framework. By regarding the structure of wafer defect maps as point sets, polygon-based features are extracted and used in a global cluster flow. Moreover, the new generation AI techniques based on large pre-trained foundation models are introduced as our training paradigm. The experimental results demonstrate the superiority of our framework. We expect our effort will shed light on the cutting-edge IC manufacturing research with emerging AI techniques.

Acknowledgment

This work is sponsored by Shanghai Pujiang Program (Project No. 22PJ1410400).

References

[1] A. A. Ezzat, S. Liu, D. S. Hochbaum, and Y. Ding, "A graph-theoretic approach for spatial filtering and its impact on mixed-type spatial pattern recognition in wafer bin maps," *IEEE Transactions on Semiconductor Manufacturing (TSM)*, 2021.

[2] H. Geng, F. Yang, X. Zeng, and B. Yu, "When wafer failure pattern classification meets few-shot learning and self-supervised learning," in *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2021.

[3] H. Geng, Q. Sun, T. Chen, Q. Xu, T.-Y. Ho, and B. Yu, "Mixed-type wafer failure pattern recognition," in *IEEE/ACM Asia and South Pacific Design Automation Conference (ASPDAC)*, 2023.

[4] J. Wang, C. Xu, Z. Yang, J. Zhang, and X. Li, "Deformable convolutional networks for efficient mixed-type wafer defect pattern recognition," *IEEE Transactions on Semiconductor Manufacturing (TSM)*, 2020.

[5] Y. Wei and H. Wang, "Mixed-type wafer defect recognition with multi-scale information fusion transformer," *IEEE Transactions on Semiconductor Manufacturing (TSM)*, 2022.

[6] Z. Liu, Y. Lin *et al.*, "Swin transformer: Hierarchical vision transformer using shifted windows," in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[7] H. Edelsbrunner and E. P. Mücke, "Three-dimensional alpha shapes," *ACM Transactions on Graphics (TOG)*, 1994.

[8] T. Su, W. Wang, Z. Lv, W. Wu, and X. Li, "Rapid delaunay triangulation for randomly distributed point cloud data using adaptive hilbert curve," *Computers and Graphics*, 2016.

[9] X. Ma, Y. Zhou, H. Wang, C. Qin, B. Sun, C. Liu, and Y. Fu, "Image as set of points," in *International Conference on Learning Representations (ICLR)*, 2023.

[10] X. Ma, C. Qin, H. You, H. Ran, and Y. Fu, "Rethinking network design and local geometry in point cloud: A simple residual mlp framework," *International Conference on Learning Representations (ICLR)*, 2022.

[11] A. Kirillov, E. Mintun *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.

[12] G.-S. Xia, J. Hu, F. Hu, B. Shi, X. Bai, Y. Zhong, L. Zhang, and X. Lu, "Aid: A benchmark dataset for performance evaluation of aerial scene classification," *IEEE Transactions on Geoscience and Remote Sensing (TGRS)*, 2017.