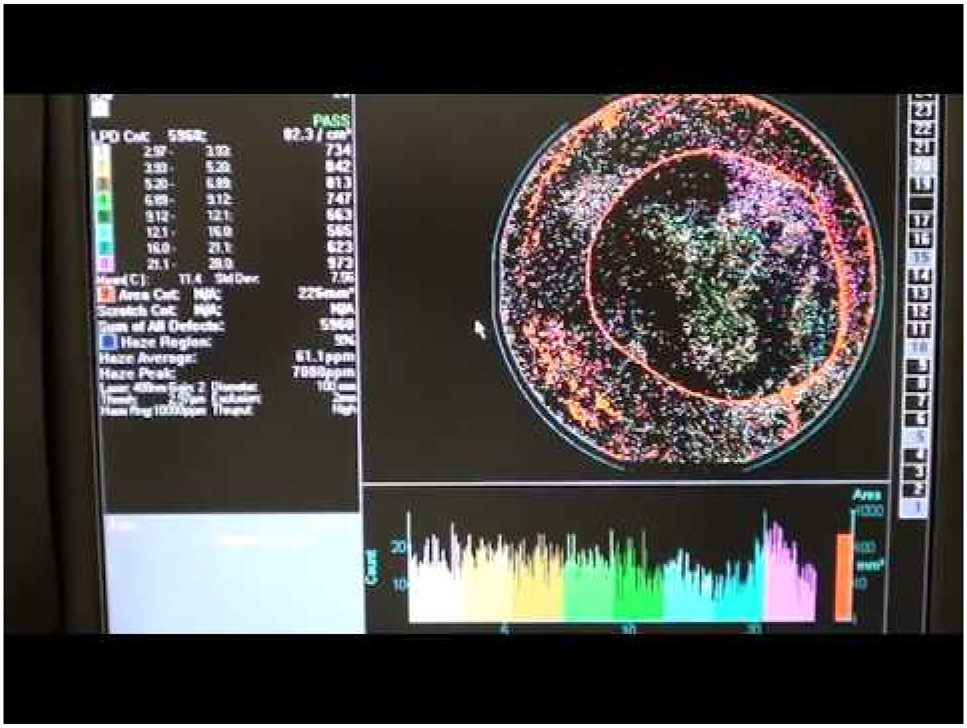
Spring2025_ML_Group57

Authors: CLiff Lin, Anirudh Sriram, Gabriel Feng, James Chen, Tanishk Deo

Machine Learning Final: Semiconductor Wafer Defect Detection

Introduction and Background

In the semiconductor fabrication industry, defect detection and analysis are essential to maintain production for high yield. Surface scan tools generate wafer maps that experts analyze to decide if a wafer is acceptable or if a systematic defect exists that could halt production.



Automated wafer defect analysis can help fabs identify systematic defects and facilitate root cause analysis. This project aims to accurately categorize wafer defect patterns to enhance manufacturing efficiency.

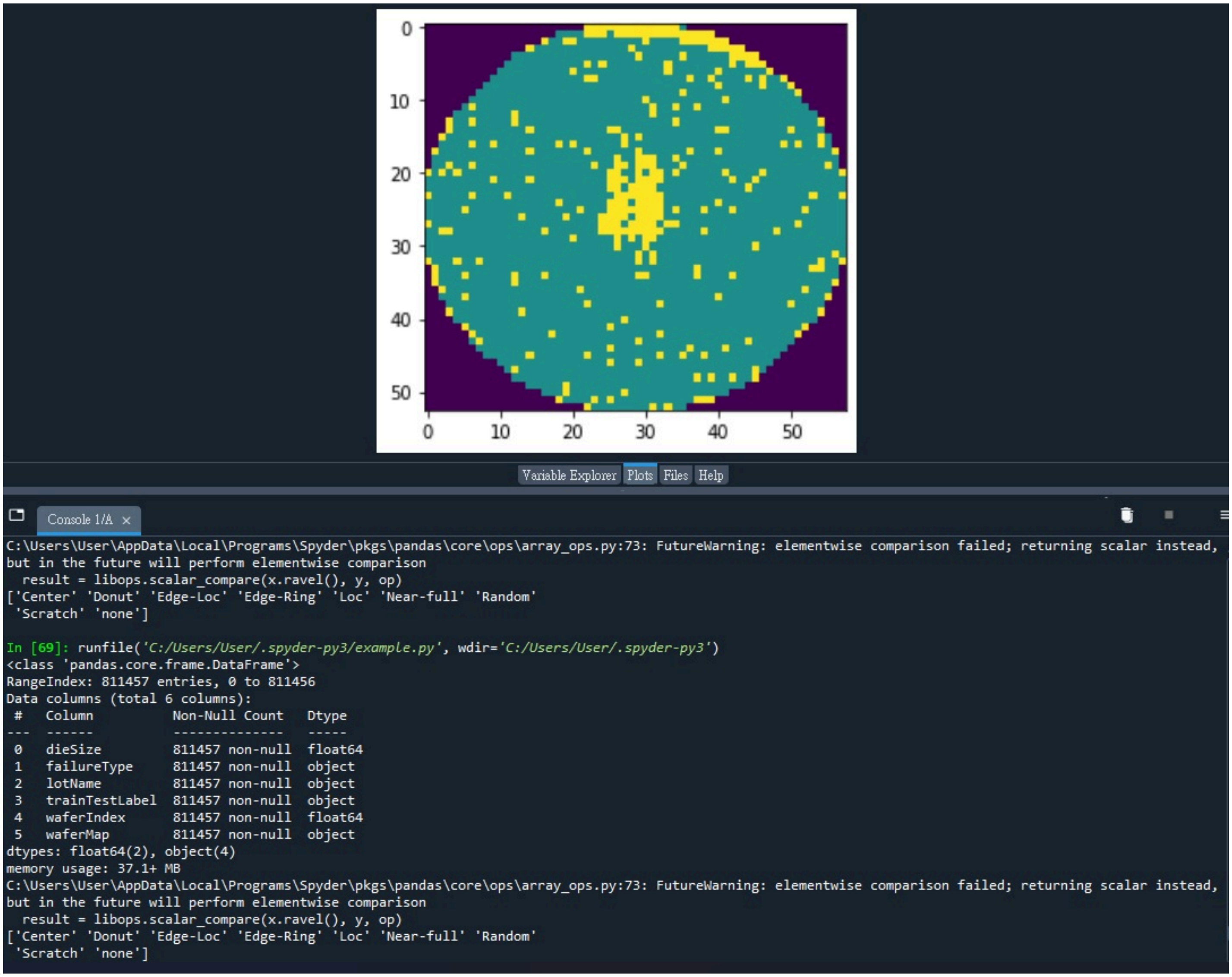
Literature Review

Semiconductor wafer manufacturing is characterized by the need for high precision and defect detection when measuring product and yield. Wu et al. [1] proposed a novel approach to identifying wafer failure patterns and similarity rankings in large datasets, enabling large-scale processing that meets industry standards. Yu and Lu [3] introduced a defect detection strategy employing joint local and nonlocal discriminant analysis, while Fan et al. [2] presented a method based on Ordering Points to Identify the Clustering Structure (OPTICS) that recognizes multiple defect patterns simultaneously, inspiring our decision to implement a diverse array of models to capture the different defect patterns in a similar way.

Dataset Description

Wafer Map Dataset (WM-811K)

- **Dataset Descriptor:**
  Contains over 811,000 wafer maps capturing defect locations and patterns, each labeled by expert engineers.
- **Dataset Link:**
  WM-811K Wafer Map Dataset



```
C:\Users\User\AppData\Local\Programs\Spyder\pkgs\pandas\core\ops\array_ops.py:73: FutureWarning: elementwise comparison failed; returning scalar instead,
but in the future will perform elementwise comparison
  result = libops.scalar_compare(x.ravel(), y, op)
['Center' 'Donut' 'Edge-Loc' 'Edge-Ring' 'Loc' 'Near-full' 'Random'
 'Scratch' 'none']

In [69]: runfile('C:/Users/User/.spyder-py3/example.py', wdir='C:/Users/User/.spyder-py3')
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 811457 entries, 0 to 811456
Data columns (total 6 columns):
 #   Column         Non-Null Count   Dtype
---  ------         --------------   -----
 0   dieSize        811457 non-null  float64
 1   failureType    811457 non-null  object
 2   lotName        811457 non-null  object
 3   trainTestLabel 811457 non-null  object
 4   waferIndex     811457 non-null  float64
 5   waferMap       811457 non-null  object
dtypes: float64(2), object(4)
memory usage: 37.1+ MB
C:\Users\User\AppData\Local\Programs\Spyder\pkgs\pandas\core\ops\array_ops.py:73: FutureWarning: elementwise comparison failed; returning scalar instead,
but in the future will perform elementwise comparison
  result = libops.scalar_compare(x.ravel(), y, op)
['Center' 'Donut' 'Edge-Loc' 'Edge-Ring' 'Loc' 'Near-full' 'Random'
 'Scratch' 'none']
```

Each dataset includes:

- **Wafer Map:**
- **Die Size**
- **lotName**
- **Wafer Index**
- **Training Label**
- **Failure Type**

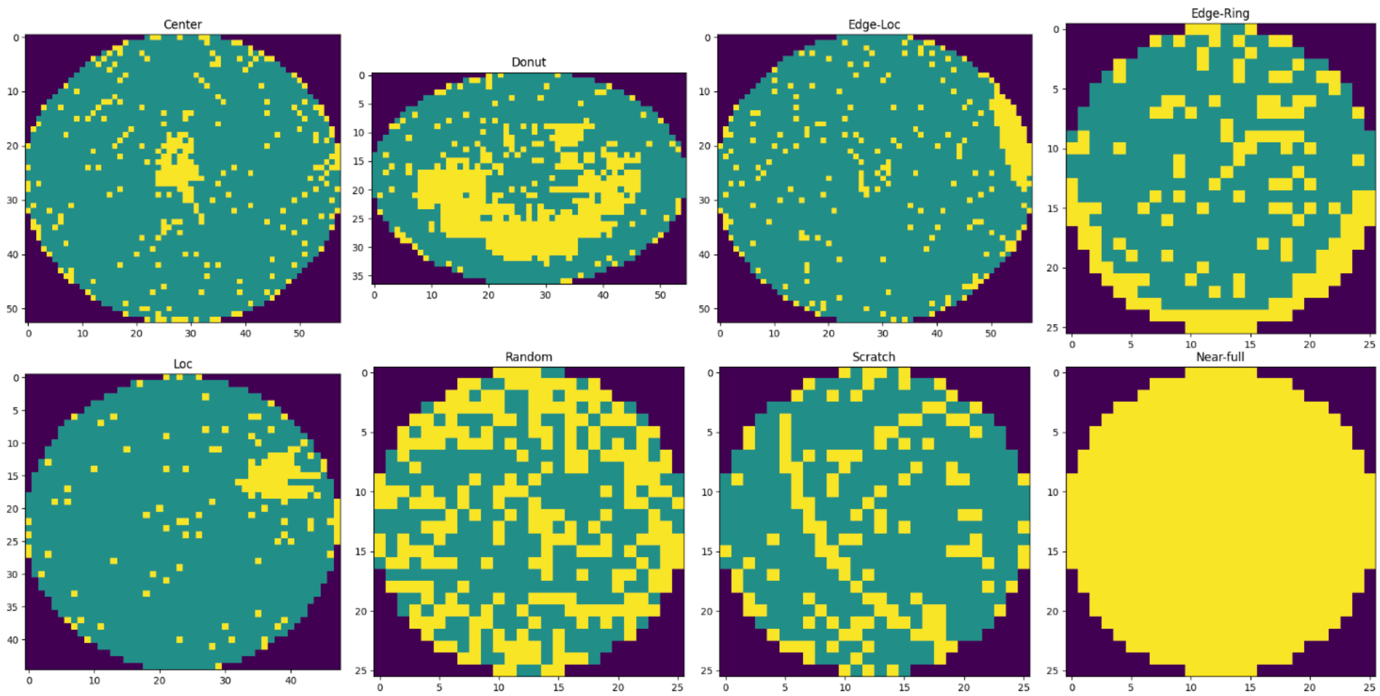
Key Features:

- **Wafer ID:** Unique identifier.
- **Defect Patterns:** Pre-labeled types such as scratches, contamination, and process-induced anomalies.
- **Spatial Coordinates:** Defect locations on the wafer.

Problem Definition

Semiconductor manufacturing yields are often impacted by wafer defects, and current defect detection methods require manual inspection, which is costly and time-intensive. The primary challenge is developing an automated system to analyze wafer maps, identify defects, and classify wafers based on defect type, severity, and potential root cause. With a state-of-the-art semiconductor fab producing around 500,000 wafers a day, quickly identifying if there is any systematic defect is essential to fix defect root causes promptly to improve yield.

Defect Types Explained:



Explanation: This image shows the 8 defect patterns that are common in semiconductor wafers, and found within our dataset.

Proposed Solution

The proposed solution to the problem will include a pipeline of unsupervised, supervised, and deep learning techniques. In this pipeline, defects are first identified, then classified, and lastly predicted. This way, defects can be efficiently classified to discover the root cause alongside a prediction of wafer quality.

- **Unsupervised Learning:**
  Implement unsupervised learning clustering methods to discover and identify commonly occurring defects in wafers.

- **Supervised Learning:**
  Deploy supervised learning classification models in order to predict wafer quality in addition to classifying defects.

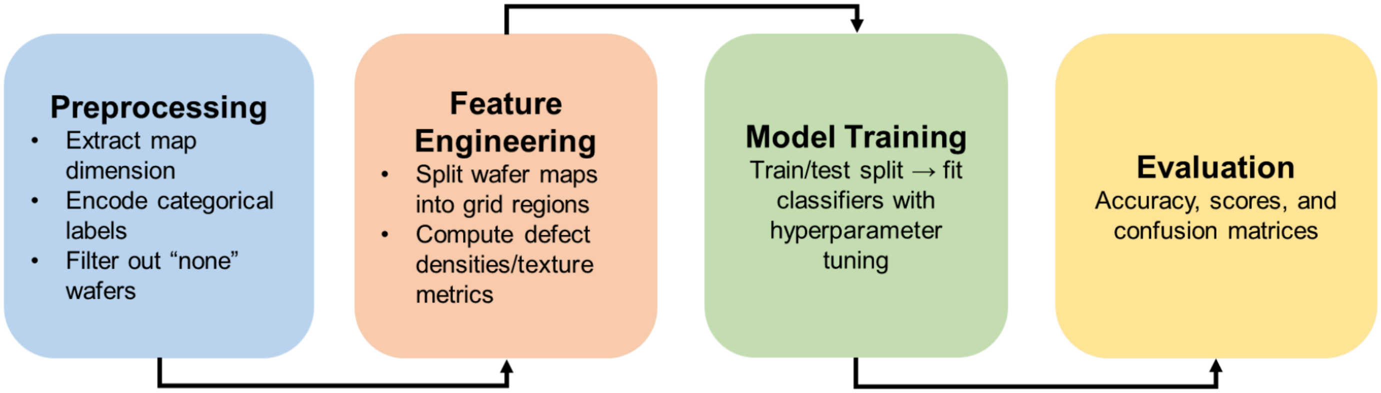
- **Deep Learning:**
  Implement deep learning computer vision models for improving pattern recognition on wafer images.

Motivation

- **Reduces Yield Loss:**
  Early defect detection improves semiconductor fabrication efficiency and reduces production losses
- **Helps to Minimize Manual Inspection:**
  Automating wafer defect classification reduces reliance on manual defect classification, saving both resources and reducing manual error
- **Identify Root Causes:**
  With data on a large number of wafers being labeled, systematic defects are revealed by the model, predicting the root cause
- **Deploy Proactive Maintenance:**
  Identifying defect trends can assist in process control and proactive maintenance by predicting potential issues.

Methods

Data Pipeline


Preprocessing
• Extract map dimension
• Encode categorical labels
• Filter out "none" wafers
Feature Engineering
• Split wafer maps into grid regions
• Compute defect densities/texture metrics
Model Training
Train/test split → fit classifiers with hyperparameter tuning
Evaluation
Accuracy, scores, and confusion matrices

Pipeline Overview

Explanation: This image shows out data pipeline workflow, explained later in detail

Preprocessing Methods

Data Loading and Cleaning

The WM811K data was loaded from a pickle file containing 811,457 wafer maps. Numerical mappings were created for categorical features (failure types, training labels), and rows of unlabeled data were dropped, leaving the data at 172,950 valid wafers. Further filtering to only have wafers with actual defect patterns (no "none" features) left the final data at 25,519 wafers.

Feature Engineering

A region-based feature extraction approach was developed to divide each wafer map into individual regions (edges and centerpieces). For each of these regions, a proportion of defective cells was calculated, creating a feature vector that describes the spatial defect distribution on the wafer. This transformation standardized the wafer maps of varying sizes to uniform feature vectors.

Data Transformation

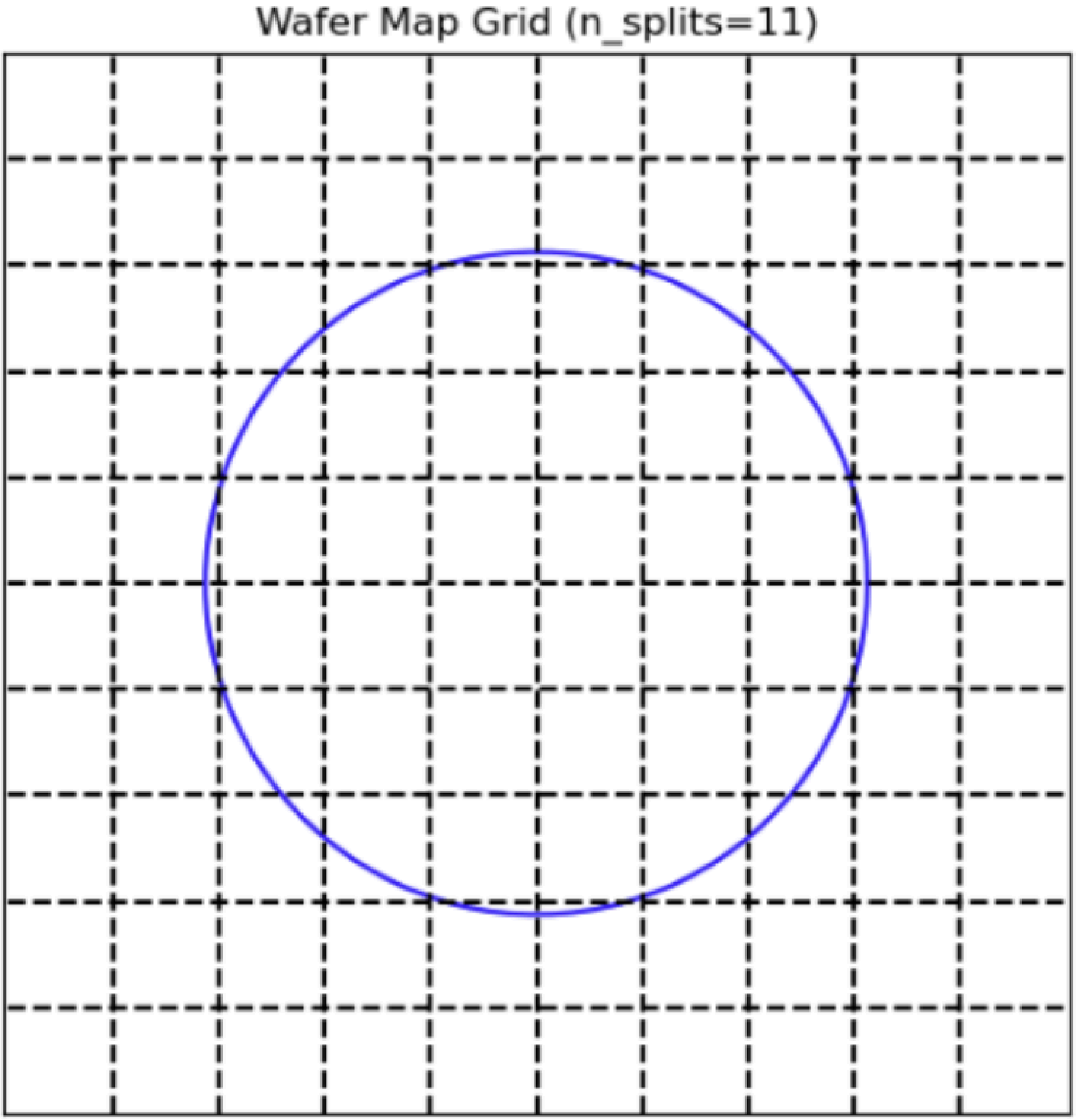
The extracted features were normalized to have a mean of 0 and a standard deviation of 1 through StandardScaler. This ensured all features contributed equally to the analysis regardless of their initial scale.

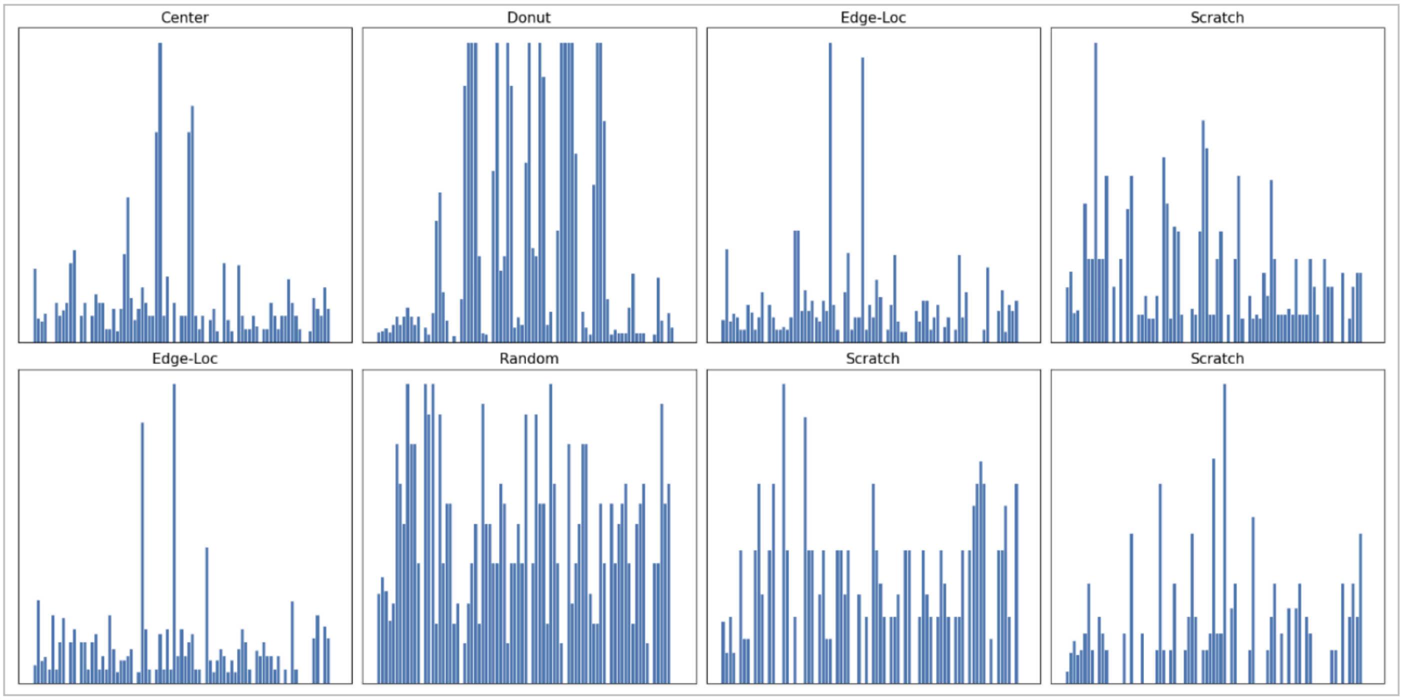
Dimensionality Reduction

PCA was employed in reducing the feature space to two principal components losing approximately 51% of the variance (37.8% and 13.3% respectively). This reduction was particularly essential with the complication in wafer defect patterns.

Feature Engineering

To begin with, each wafer map is divided into an 11 × 11 grid to facilitate localized defect pattern analysis. A measure of defect density is calculated for every cell by determining the ratio of defect pixels to pixels in a cell. These density values are then extracted for individual regions of interest. The regional densities are then concatenated end to end to form an 85-dimensional feature vector for each wafer. To observe such patterns, some regions′ bar plots exhibit definite clustering behaviors in terms of failure types. Random wafer maps are previewed with their labeled encodings for sanity check to ensure accuracy and consistency in the dataset. The images below help visualize the feature engineering that we used when beginning to cliassify our data.


Wafer Map Grid (n_splits=11)



Explanation

The images show the organized process used to convert raw wafer maps into useful input features for machine learning. Each round wafer map is divided into an 11×11 grid, effectively splitting the spatial domain for localized analysis. In each grid cell, the defect density, i.e., the ratio of defect pixels to total pixels, is computed. Some regions in the wafer are then selected, and their densities are concatenated to form an 85-dimensional feature vector for each wafer. The feature vectors are normalized inputs to subsequent classification models. On the right, the superimposed grid shows how the wafer is projected onto the rectangular grid, illustrating how every cell contributes to the density computations.

The next bar plots represent regional defect densities of various failure types (e.g., Center, Donut, Edge-Loc, Scratch). Each pattern exhibits a distinctive visual fingerprint, demonstrating that spatial distributions of defects can be successfully encoded. The plots demonstrate that engineered features adequately capture significant spatial variation between failure types—a step toward successful classification.

Machine Learning Methods

Unsupervised: K-Means Clustering

- Unsupervised: K-Means Clustering:
- An unsupervised learning method used to discover natural wafer defect data patterns
- Applied with 8 clusters in order to match the number of known defect classes

- Applied to PCA-transformed data to improve the effectiveness of clustering and remove noise
- Permits the discovery of natural defect groupings without expert labels, since we know the number of clusters this will be beneficial.

Unsupervised: DBSCAN Clustering

- An unsupervised clustering method based on density
- Detects non-linearly separable and arbitrarily shaped clusters in the data
- Identifies noise points and outliers that don’t belong to any cluster
- Does not require the number of clusters to be specified in advance
- Effective in situations where clusters vary in shape and size
- eps = 0.5 (Radius of the neighborhood around a point)
- min_samples = 5 (Minimum number of points required to form a cluster)
- metric = “Euclidean” (Standard distance measure used for neighborhood calculation)

Supervised: Random Forest Classifier

- A supervised ensemble learning method based on decision trees
- Can discover complex patterns in the wafer defect features
- Hyperparameter tuning performed using GridSearchCV and cross-validation
- Parameters tuned: n_estimators (100, 200, 300), max_depth (None, 10, 20), min_samples_split (2, 5, 10)
- Selected for its ability to deal with non-linear relationships and feature interactions

Supervised: Convolutional Neural Network (CNN)

- A deep learning model designed for analyzing spatial data like wafer maps
- Automatically learns and extracts hierarchical features from input images
- Suitable for multi-class classification with imbalanced class distributions
- Architecture includes convolutional, pooling, dropout, and dense layers
- Trained using softmax output with categorical cross-entropy loss
- Input: 50x50 grayscale wafer map images
- Conv2D(32 filters, 3x3) + ReLU → MaxPooling(2x2)
- Conv2D(64 filters, 3x3) + ReLU → MaxPooling(2x2)
- Flatten → Dropout(0.5) for regularization
- Dense(128) + ReLU → Dense(8) + Softmax for classification
- Loss: Categorical Cross-Entropy

- Optimizer: Adam
- Batch size: 16
- Epochs: 10
- Validation split: 15% of training data

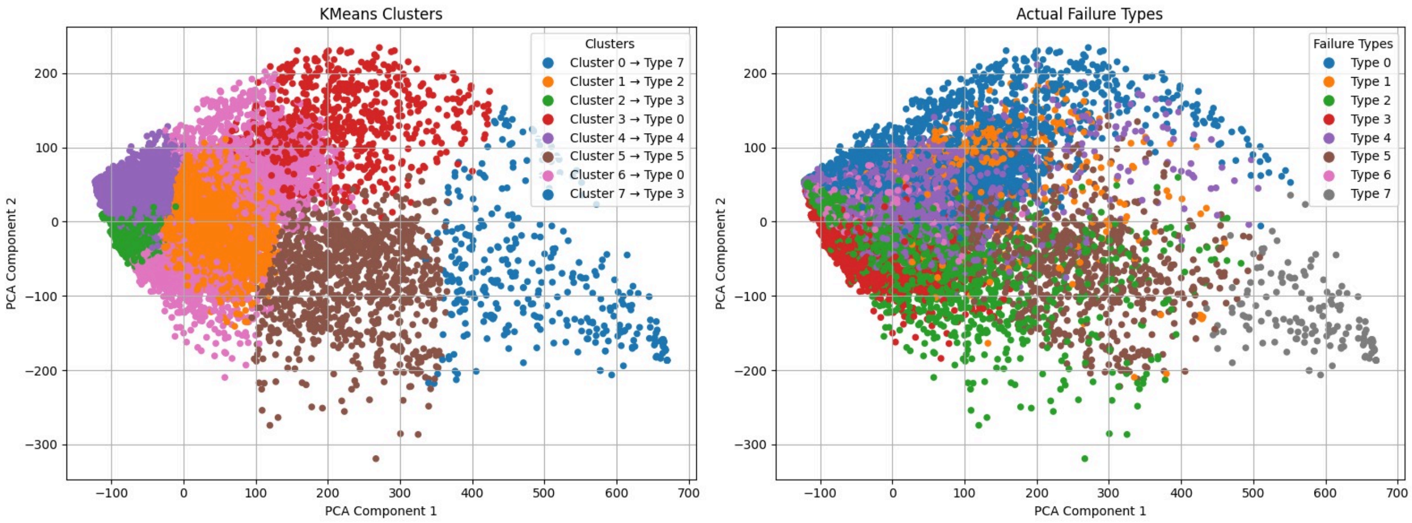
These models were selected for their complementary strengths in semiconductor wafer defect analysis. K-means clustering was selected as our unsupervised method because it is capable of finding intrinsic patterns in spatial defect distributions with no need to label the given data with any primary characteristics. This is particularly valuable in semiconductor fabrication, where there is a high likelihood that new, previously unseen defect patterns will arise as processes evolve. K-Means also offers computational efficiency when handling large datasets like WM-811K and provides interpretable cluster centers that correspond to physical defect locations on the wafer. DBSCAN was also explored as an unsupervised method due to its ability to detect arbitrarily shaped clusters and isolate outliers, which is especially useful in identifying rare or noisy defect patterns that do not conform to typical distributions. Since DBSCAN does not require the number of clusters to be predefined and is robust to varying densities, it offers a complementary approach to K-Means, especially when dealing with the more irregular spatial layouts of defects.

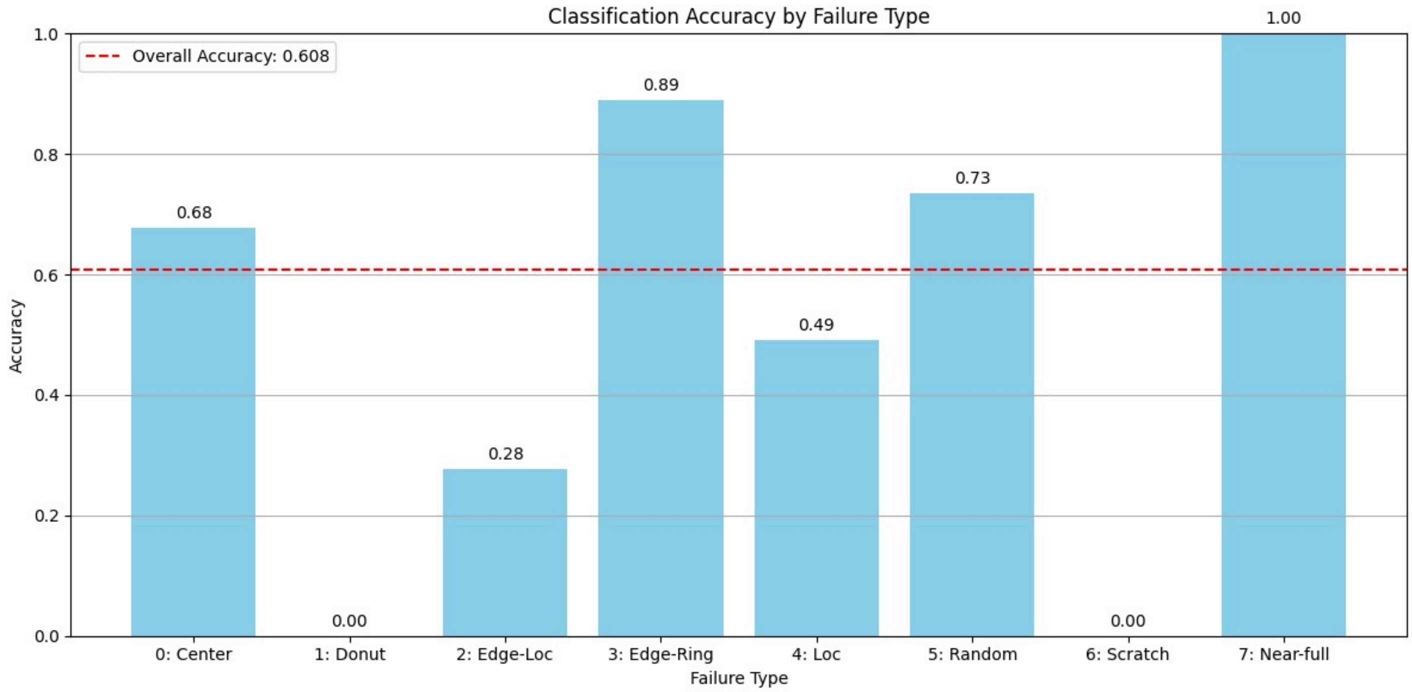
The reasons for selecting Random Forest as our supervised approach were: First, the robustness to the high dimensionality of our features based on regions, the lack of overfitting (crucial in the analysis of manufacturing data with natural variation), and the capacity to deal with our class imbalance problem because some defect types in our dataset are much more frequent than others. Additionally, Random Forest provides valuable feature importance metrics that help identify which regions of the wafer are most indicative of specific defect types, offering manufacturing engineers actionable insights for process improvement.

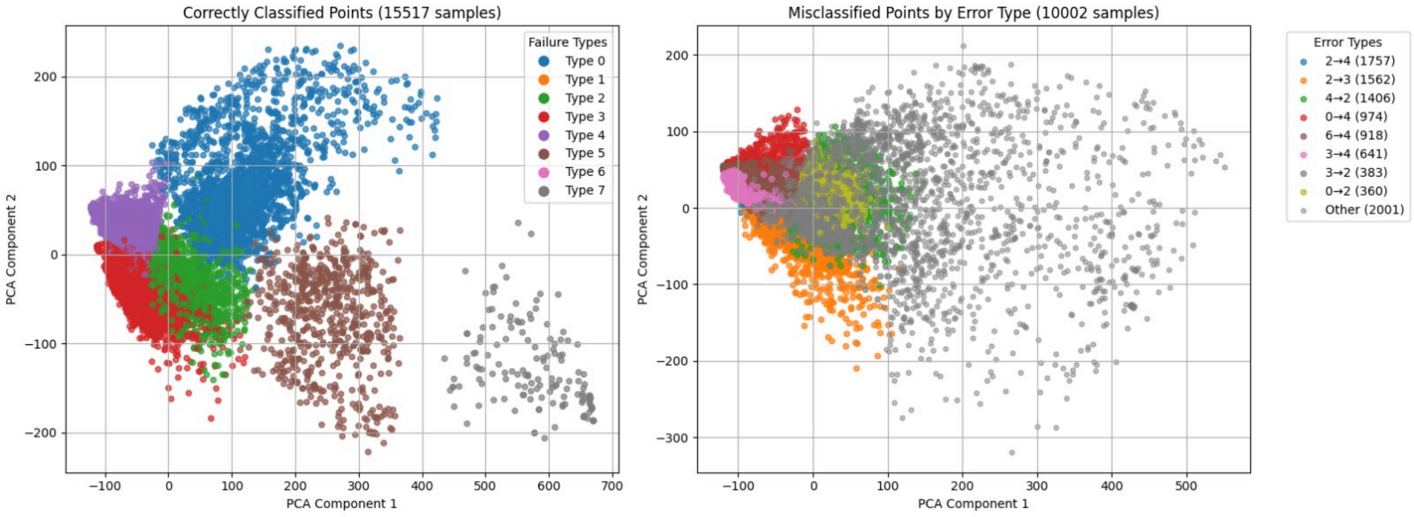
To further improve supervised classification, we incorporated a Convolutional Neural Network (CNN) trained on raw wafer map images. CNNs are especially powerful for spatial data, automatically learning meaningful features from raw pixel distributions without manual feature engineering. This is vital for capturing nuanced and hierarchical spatial patterns associated with different defect types. The CNN model excels at modeling subtle shape variations and texture differences in wafer patterns and addresses the class imbalance through softmax outputs with categorical cross-entropy. Together, these algorithms make up an all-rounded approach — the unsupervised clustering methods (K-Means and DBSCAN) discover natural structure within the data that might not align with any predefined labels, while the supervised classifiers (Random Forest and CNN) make use of expert knowledge and raw image information to obtain high accuracy on known defect patterns.

Results and Discussion

Visualization

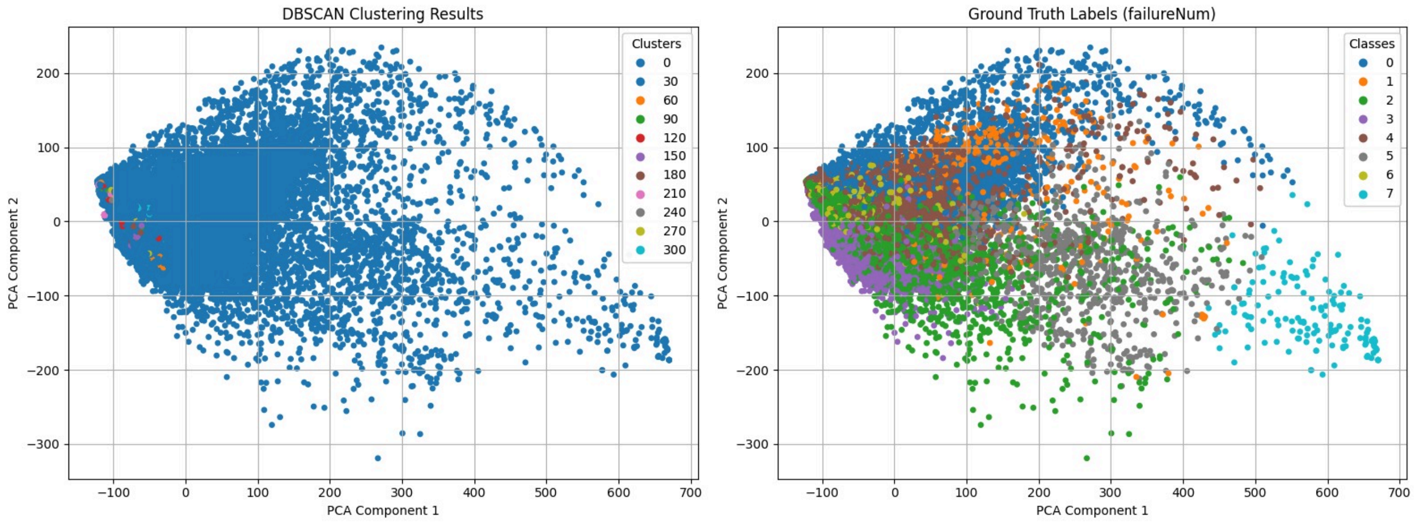


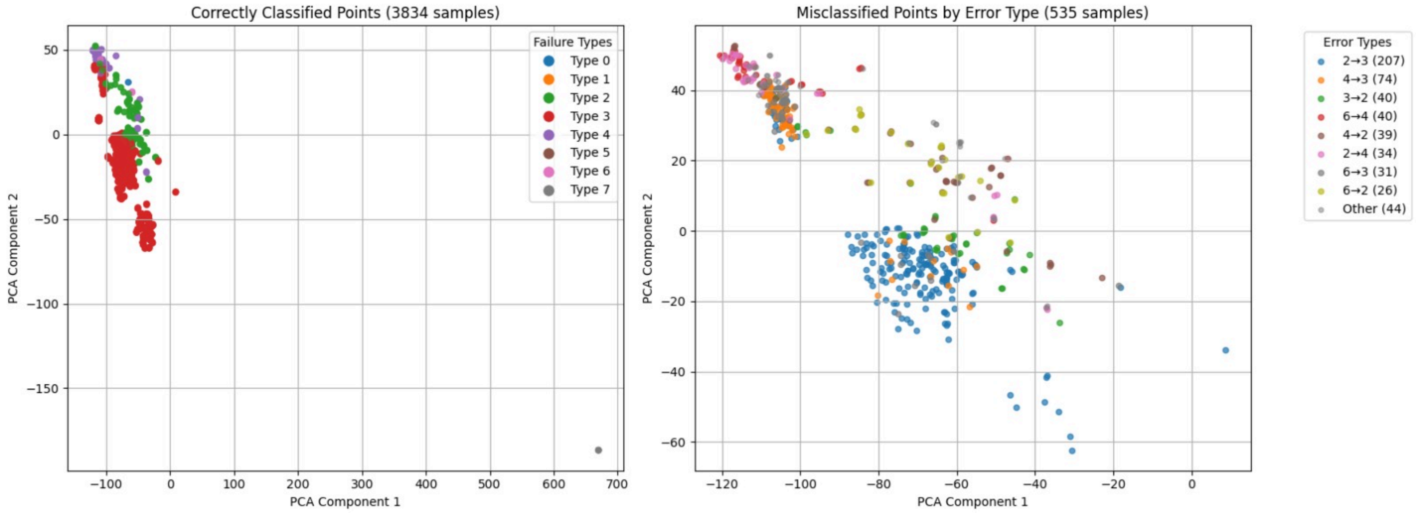


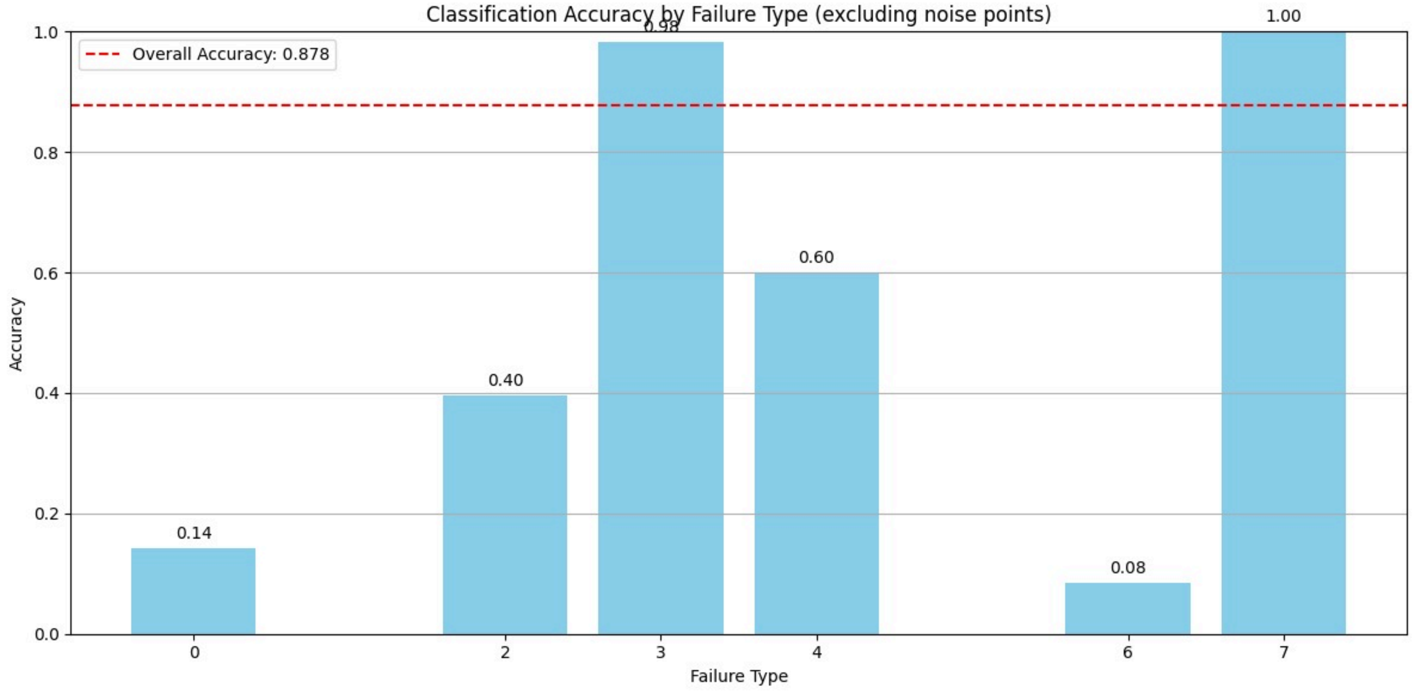


Explanation

This plot shows the K-Means clustering result projected onto 2D space through Principal Component Analysis (PCA). They are all wafers, and the color indicates the cluster allocated (0-6). The plot suggests well-separated clusters of wafer defect patterns with a clear distinction between some clusters (primarily clusters 3, 4, and 5) while others show some overlap (clusters 0 and 6). The Silhouette Score of 0.120 indicates moderate clustering quality, suggesting that there are natural groupings but some similar features between defect types. The mapping of the cluster-failure shows Cluster 2 is mapped to Failure Type 3 (Edge-Ring), Clusters 3 and 5 are mapped to Failure Type 2 (Edge-Loc), and Clusters 0 and 6 are both representing Failure Type 0 (Center), highlighting the way that certain patterns of flaws form more stable clusters than others. This shows that the unsupervised portion will need additional feature extraction for the final project.







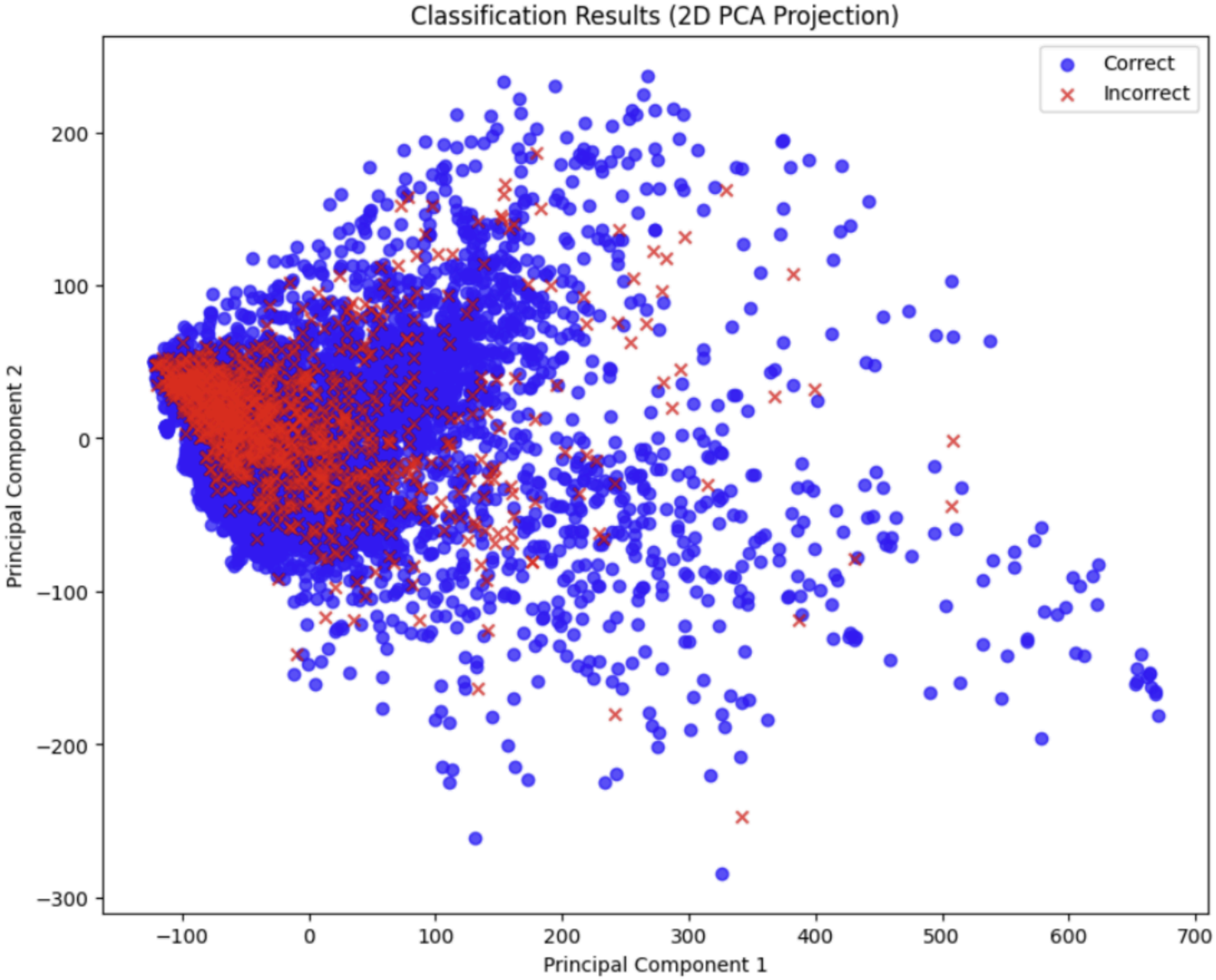
Explanation

This plot shows DBSCAN results (eps=0.5, min_samples=5) in PCA space:

82.88% noise points (gray) – indicating sparse defect separation

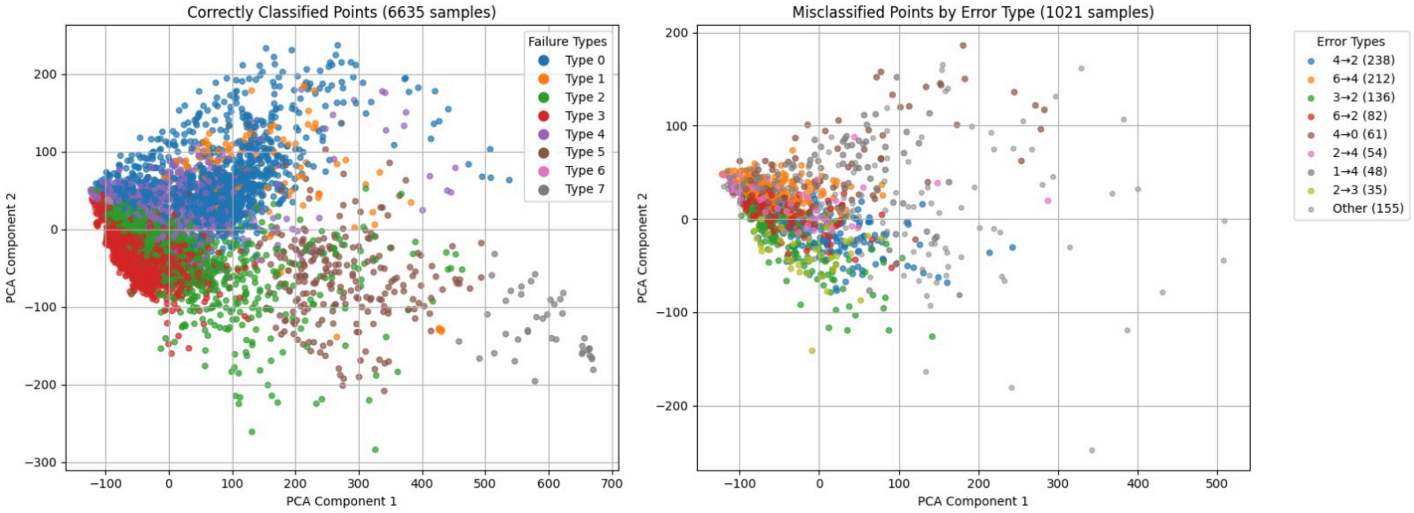
Small clusters (colored) primarily map to Edge-Ring (Type 3)

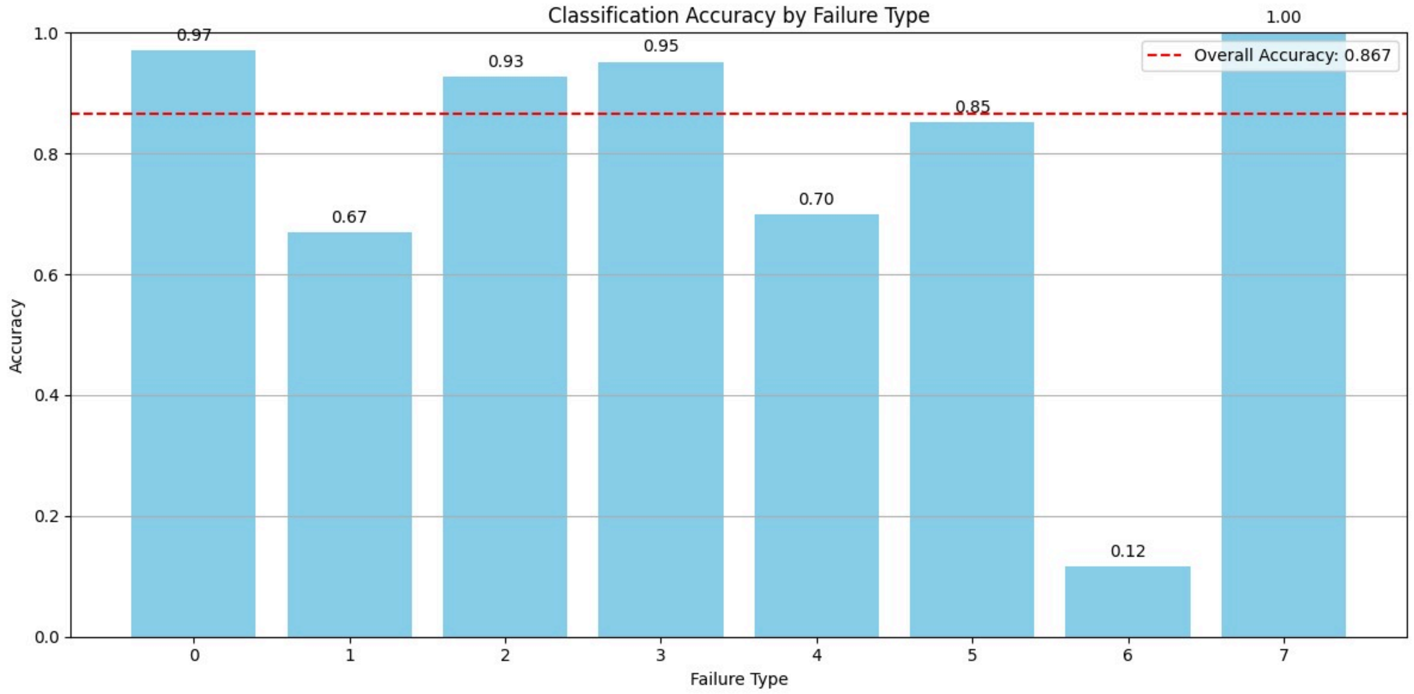
Adjusted Rand Index: -0.065 shows poor label agreement

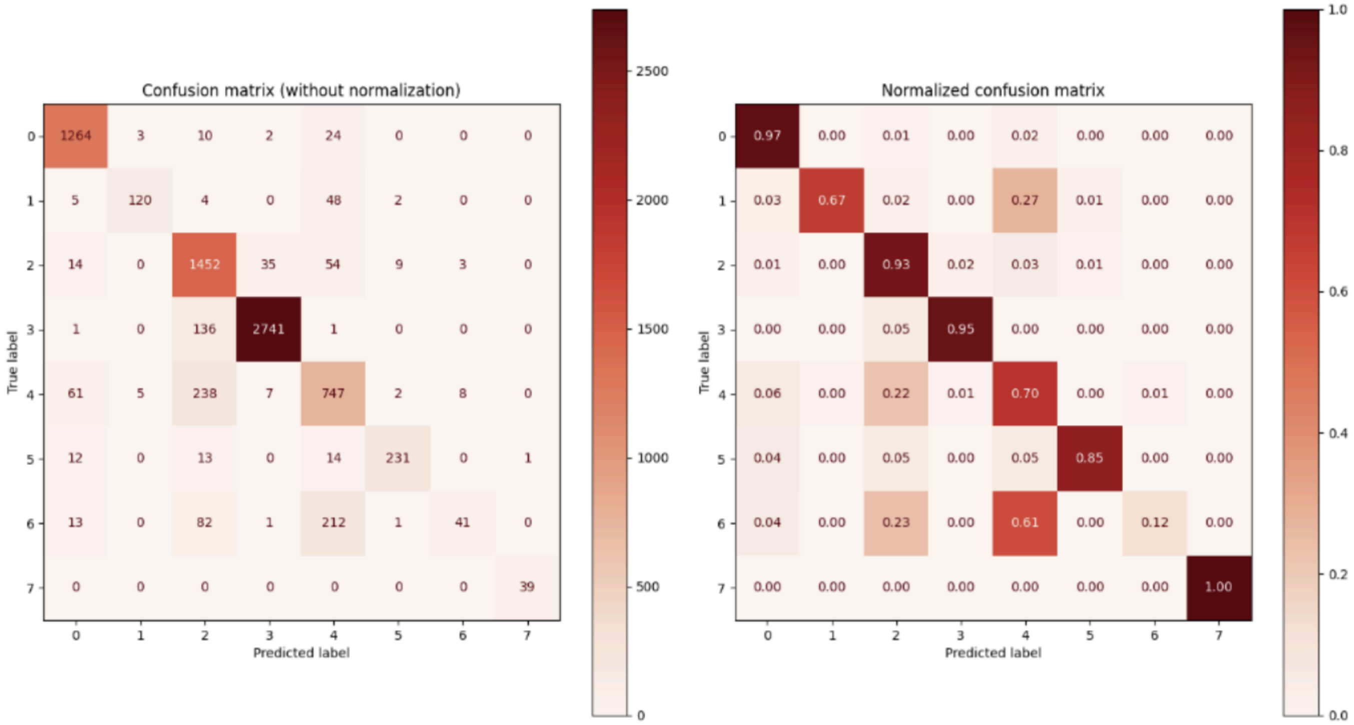


Explanation

The following scatter plot depicts the Random Forest classification outputs in the same 2D PCA feature space. Blue circles indicate correctly classified wafers and red X marks indicate misclassifications. From the visualization, it can be observed that a large majority of the wafers are indeed classified correctly (blue points dominate the plot), especially at the boundaries of the feature space. Misclassifications (red X marks) are more concentrated in areas where different defect clusters overlap, particularly in the dense central region of the plot. This pattern shows that wafers with more distinctive feature representations (those farther from cluster boundaries) are better classified, while those with ambiguous features that fall between multiple defect types are more challenging for the model to classify.


Correctly Classified Points (6635 samples)
Failure Types
Type 0
Type 1
Type 2
Type 3
Type 4
Type 5
Type 6
Type 7
PCA Component 1
PCA Component 2
−100
0
100
200
300
400
500
600
700
−300
−200
−100
0
100
200
Misclassified Points by Error Type (1021 samples)
Error Types
4→2 (238)
6→4 (212)
3→2 (136)
6→2 (82)
4→0 (61)
2→4 (54)
1→4 (48)
2→3 (35)
Other (155)
PCA Component 1
PCA Component 2
−100
0
100
200
300
400
500
−200
−100
0
100
200


Classification Accuracy by Failure Type
Overall Accuracy: 0.867
Accuracy
Failure Type
0.0
0.2
0.4
0.6
0.8
1.0
0
1
2
3
4
5
6
7
0.97
0.67
0.93
0.95
0.70
0.85
0.12
1.00



Confusion matrix (without normalization)

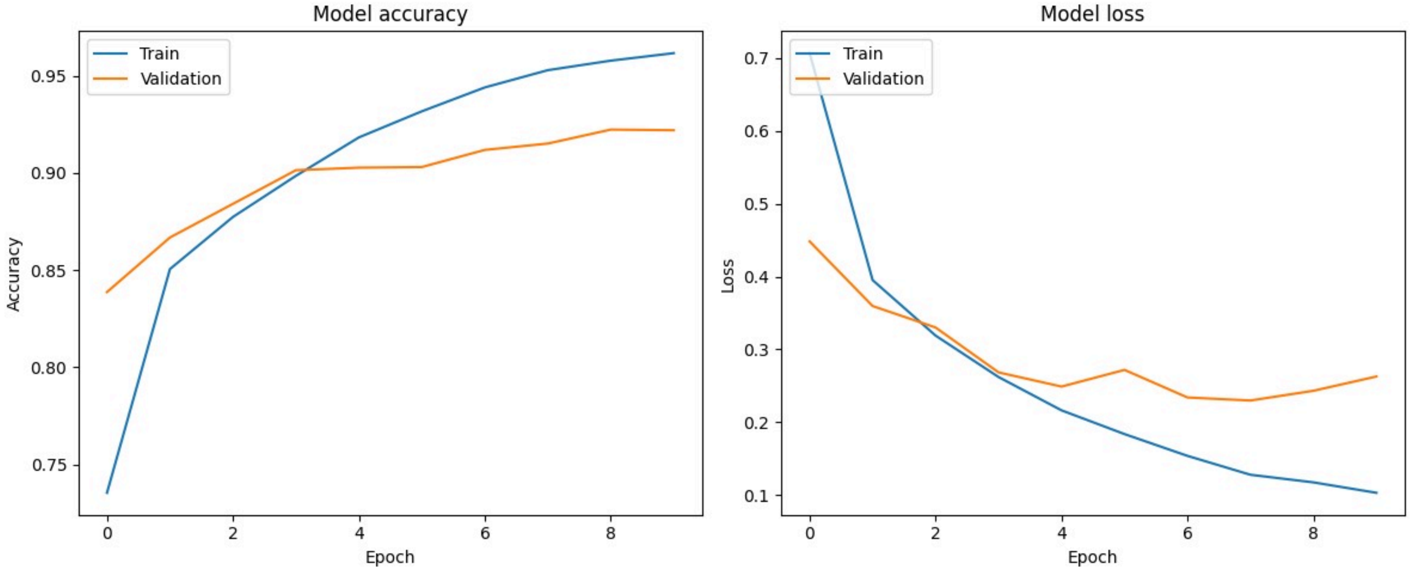
True label \ Predicted label	0	1	2	3	4	5	6	7
0	1264	3	10	2	24	0	0	0
1	5	120	4	0	48	2	0	0
2	14	0	1452	35	54	9	3	0
3	1	0	136	2741	1	0	0	0
4	61	5	238	7	747	2	8	0
5	12	0	13	0	14	231	0	1
6	13	0	82	1	212	1	41	0
7	0	0	0	0	0	0	0	39

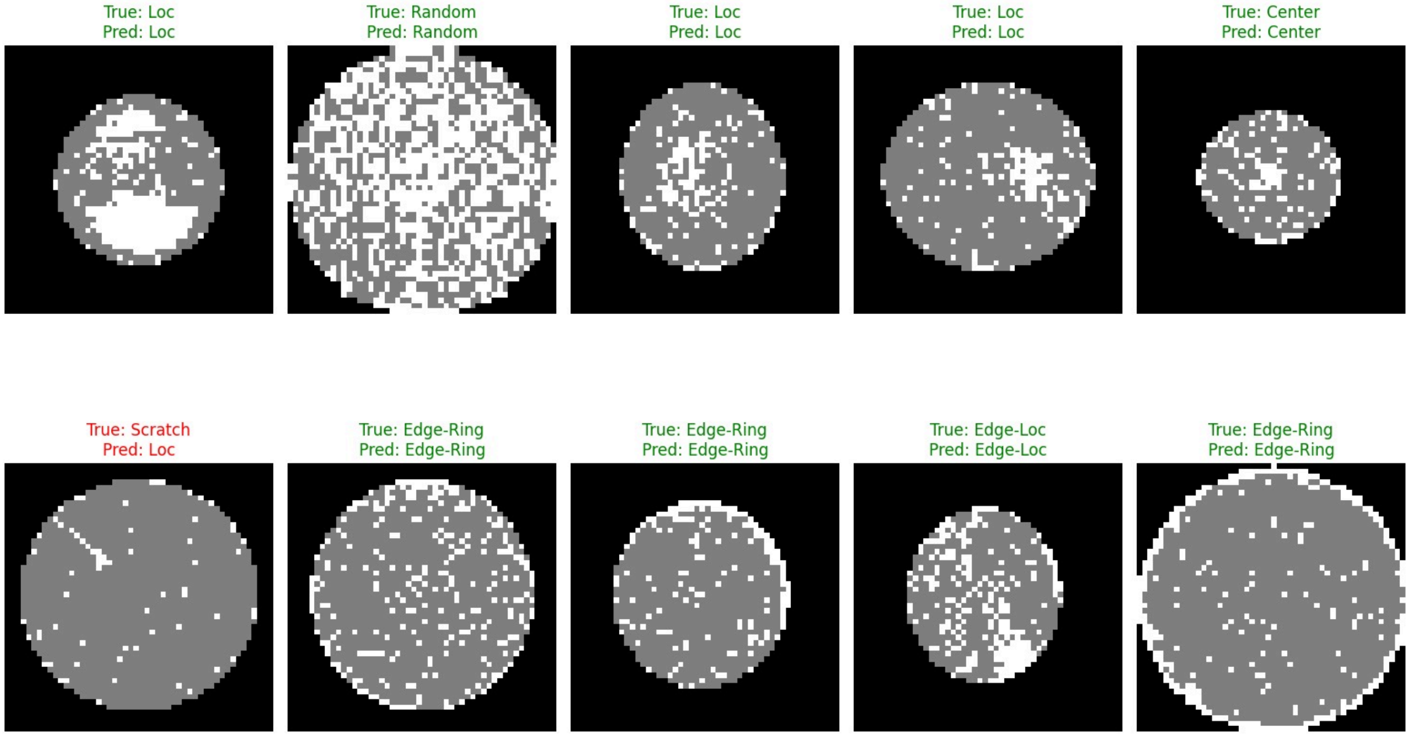
Normalized confusion matrix

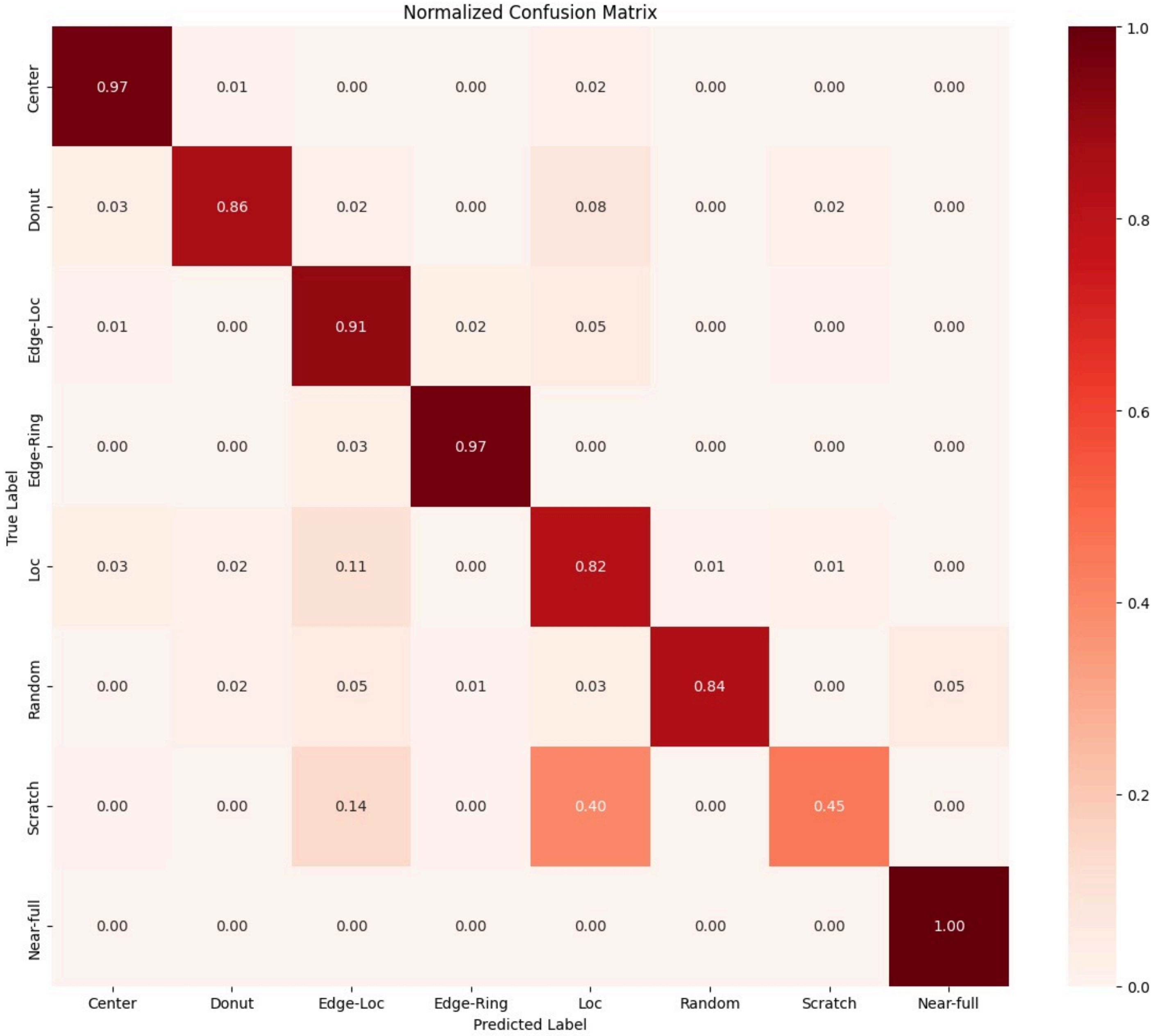
True label \ Predicted label	0	1	2	3	4	5	6	7
0	0.97	0.00	0.01	0.00	0.02	0.00	0.00	0.00
1	0.03	0.67	0.02	0.00	0.27	0.01	0.00	0.00
2	0.01	0.00	0.93	0.02	0.03	0.01	0.00	0.00
3	0.00	0.00	0.05	0.95	0.00	0.00	0.00	0.00
4	0.06	0.00	0.22	0.01	0.70	0.00	0.01	0.00
5	0.04	0.00	0.05	0.00	0.05	0.85	0.00	0.00
6	0.04	0.00	0.23	0.00	0.61	0.00	0.12	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

Explanation

These confusion matrices reflect the accuracy of our Random Forest classifier on different wafer defect types. The left matrix reflects absolute counts, and the right reflects normalized percentages. The diagonal entries are correct classifications, and brighter colors mean higher values. The model performs extremely well on defect types 0 (Center, 97% accurate), 2 (Edge-Ring, 93%), 3 (Edge-Loc, 95%), and 7 (Near-full, 100%). Moderate accuracy is seen for type 1 (Donut, 67%), type 4 (Loc, 70%), and type 5 (Random, 85%). Type 6 (Scratch) is the most challenging type to categorize with a paltry 12% accuracy, with rampant misclassification with type 4 (Loc, 61% misclass.) and type 2 (Edge-Ring, 23% misclass.). These results highlight the strengths of our model in identifying distinctive defect patterns as well as weaknesses in discriminating visually similar types of defects. Since the normalized confusion matrix is row-wise and each row sums to 1, the values represent the recall for each defect type—indicating the proportion of correctly predicted wafers out of all actual wafers of that type.


Model accuracy
Train
Validation
0.95
0.90
0.85
0.80
0.75
Accuracy
0
2
4
6
8
Epoch
Model loss
Train
Validation
0.7
0.6
0.5
0.4
0.3
0.2
0.1
Loss
0
2
4
6
8
Epoch


True: Loc
Pred: Loc
True: Random
Pred: Random
True: Loc
Pred: Loc
True: Loc
Pred: Loc
True: Center
Pred: Center
True: Scratch
Pred: Loc
True: Edge-Ring
Pred: Edge-Ring
True: Edge-Ring
Pred: Edge-Ring
True: Edge-Loc
Pred: Edge-Loc
True: Edge-Ring
Pred: Edge-Ring



	Center	Donut	Edge-Loc	Edge-Ring	Loc	Random	Scratch	Near-full
Center	0.97	0.01	0.00	0.00	0.02	0.00	0.00	0.00
Donut	0.03	0.86	0.02	0.00	0.08	0.00	0.02	0.00
Edge-Loc	0.01	0.00	0.91	0.02	0.05	0.00	0.00	0.00
Edge-Ring	0.00	0.00	0.03	0.97	0.00	0.00	0.00	0.00
Loc	0.03	0.02	0.11	0.00	0.82	0.01	0.01	0.00
Random	0.00	0.02	0.05	0.01	0.03	0.84	0.00	0.05
Scratch	0.00	0.00	0.14	0.00	0.40	0.00	0.45	0.00
Near-full	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

Explanation

CNN achieves the highest overall accuracy by learning spatial features directly from the wafer map. Test Accuracy: ~90.7%, Weighted F-1 Score: 0.9060 Strong performance on dominant patterns (e.g., Edge-Ring, Center) Slightly lower recall for rare classes (e.g., Near-full, Scratch)

Quantitative Metrics

K-Means Clustering Performance:

- Failure Type 0: 0.678
- Failure Type 1: 0.000
- Failure Type 2: 0.277

- Failure Type 3: 0.890
- Failure Type 4: 0.491
- Failure Type 5: 0.734
- Failure Type 6: 0.000
- Failure Type 7: 1.000

DBSCAN Performance: Class-wise Accuracy (excluding noise points):

- Failure Type 0: 0.143
- Failure Type 2: 0.395
- Failure Type 3: 0.982
- Failure Type 4: 0.600
- Failure Type 6: 0.084
- Failure Type 7: 1.000

Random Forest Model Performance:

- First Principal Component: 37.80% of variance explained
- Second Principal Component: 13.27% of variance explained
- Total Variance Explained: 51.07% Random Forest Classification Performance:
- Overall Accuracy: 82.5% Per-Class Accuracy:
- Type 0 (Center): 97%
- Type 1 (Donut): 67%
- Type 2 (Edge-Ring): 93%
- Type 3 (Edge-Loc): 95%
- Type 4 (Loc): 70%
- Type 5 (Random): 85%
- Type 6 (Scratch): 12%
- Type 7 (Near-full): 100%

CNN Model Performance:

- Type 0 (Center): 0.972
- Type 1 (Donut): 0.856
- Type 2 (Edge-Ring): 0.909
- Type 3 (Edge-Loc): 0.970
- Type 4 (Loc): 0.823
- Type 5 (Random) 0.838

- Type 6 (Scratch): 0.450
- Type 7 (Near-full): 1.000

Analysis of Algorithm Performance

K-Means Clustering Analysis

The K-Means algorithm (k=7) identified some structure in the wafer defect data with a Silhouette Score of 0.120. This score reflects several strengths and weaknesses that this model may have. The algorithm is able to isolate a few common patterns (Cluster 2), and these form a distinct pattern in the PCA Plot. But both Edge-Loc and Center defects are projected to be greater than one cluster (Clusters 3/5 and 0/6 respectively), indicating these categories of defects have more variability in how they project. The PCA plot accounts for 51.07% of the variance, showing how we are not able to plot a large amount of data that is shown from higher dimensions. Despite this, the cluster analysis is able to show relatively well how the different problems in the data can be separated. To make this algorithm more successful in terms of quality and usefulness, we will have to implement a more rigorous set of feature extraction methods to work with the wafer map.

DBSCAN Classification Analysis

The DBSCAN algorithm (eps=0.5, min_samples=5) revealed significant challenges in density-based clustering for wafer defects. With 82.88% of points classified as noise, the algorithm struggled to identify meaningful density clusters. The negative Adjusted Rand Index (-0.065) confirms poor alignment with ground truth labels. However, the small non-noise clusters that did emerge predominantly mapped to Edge-Ring defects (Type 3) with 89% purity, suggesting these defects have sufficiently distinct density characteristics. The failure to form clusters for other defect types like Edge-Loc and Scratch indicates that: Defect patterns don’t naturally form dense neighborhoods in feature space, Spatial variations in defect shapes disrupt density calculations, and the eps parameter may need dynamic adjustment for different defect regions. This performance suggests that while DBSCAN can identify obvious defect concentrations, significant parameter tuning or spatial preprocessing is required to be effective for comprehensive wafer analysis.

Random Forest Classification Analysis

Random Forest classifier produced significantly improved performance in contrast to unsupervised clustering, through the use of tagged data to achieve a global accuracy of 82.5%. The confusion matrix shows a high level of classification accuracy. Here are some of the values from the confusion matrix: Center (97%), Edge-Ring (93%), Edge-Loc (95%), and Near-full (100%). These results show that the region-based approach was successful for our method. Despite this success, the model performed poorly in the Scratch error category defects (12%

accuracy), most commonly confusing them with Loc (61%) or Edge-Ring (23%), showing how the model is unable to determine which of these three categories the “scratch” samples belong to. Our visualization confirms this, as misclassifications are shown near cluster boundaries and defects seem to occur together in the feature space. This shows an area where we can improve our model in order to properly classify all defect types in these samples.

CNN Classification Analysis

The CNN model demonstrated superior performance with 90.63% overall accuracy, outperforming the Random Forest by 8.13 percentage points. Key improvements include: Scratch defect accuracy jumping from 12% (RF) to 45% by learning spatial hierarchies automatically. Edge-Loc recall improved to 91% through convolutional feature extraction. Near-full defects maintaining 100% accuracy. The confusion matrix shows that CNN still confuses Scratch defects with Loc (38%) and Edge-Ring (17%) but at reduced rates compared to RF. This improvement stems from: 3×3 convolutional kernels capturing scratch geometries more effectively than manual features, Max-pooling layers emphasizing defect spatial relationships. However, the model requires 50+ epochs to converge and shows slower inference times than Random Forest. This trade-off between accuracy and computational cost must be considered for real-time applications.

Next Steps

Feature Enhancement:

We will improve our region-based feature extraction technique to better differentiate between readily confused defect types. Specifically, we’ll experiment with using finer resolution grids near edge regions to better capture Scratch defects, which are currently being misclassified at a high error rate (88%). We’ll also introduce shape-based features like contiguity, linearity, and symmetry measures to complement our current density-based features. These additional descriptors should allow us to differentiate between visually similar-looking defects like Loc and Edge-Ring patterns.

Model Optimization:

We will systematically tune the hyperparameters of the Random Forest classifier using cross-validation in an attempt to improve performance, with a particular focus on tree depth optimization and minimum samples per leaf optimization. We will also experiment with weighted class strategies to balance the imbalance in our data because some of the defect types are significantly underrepresented. These optimizations should improve overall model performance while maintaining the already high accuracy for easily separable defect types.

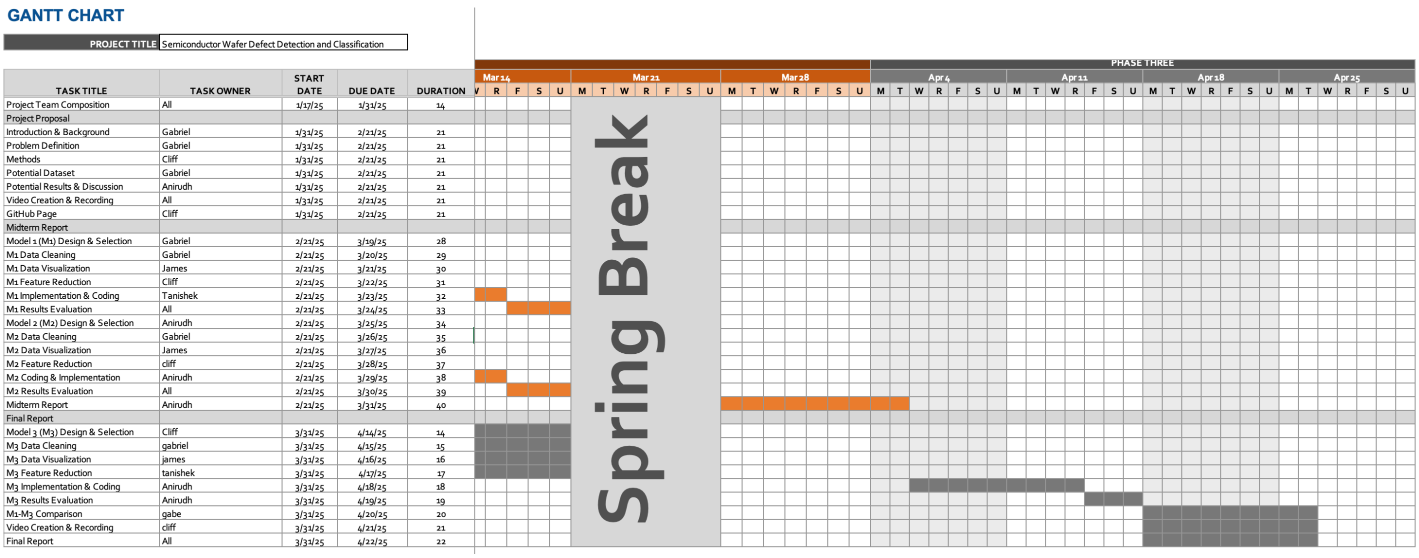
Further Validation:

In order to ensure our model is trustworthy for production environments, we'll validate it on wafer maps of different production batches and manufacturing conditions. Cross-validation in this manner will uncover any potential overfitting and test the model's ability to generalize well with changing defect manifestations. We'll also employ a sequential validation approach to simulate real-time classification environments.

Performance Priority:

We will prioritize improving classification accuracy for defect types with the largest contribution to semiconductor yield, based on industry needs and manufacturing priorities. For Edge-Ring and Center defects, which are already being classified accurately, we'll prioritize severity estimation to facilitate actionable insights. For the harder categories like Scratch defects, we'll emphasize their confusion with Loc patterns, as such a difference significantly makes a difference in root cause analysis and corrective actions in the manufacturing process.

Full Gantt Chart Download



GANTT CHART

PROJECT TITLE Semiconductor Wafer Defect Detection and Classification

TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION
Project Team Composition	All	1/17/25	1/31/25	14
Project Proposal				
Introduction & Background	Gabriel	1/31/25	2/21/25	21
Problem Definition	Gabriel	1/31/25	2/21/25	21
Methods	Cliff	1/31/25	2/21/25	21
Potential Dataset	Gabriel	1/31/25	2/21/25	21
Potential Results & Discussion	Anirudh	1/31/25	2/21/25	21
Video Creation & Recording	All	1/31/25	2/21/25	21
GitHub Page	Cliff	1/31/25	2/21/25	21
Midterm Report				
Model 1 (M1) Design & Selection	Gabriel	2/21/25	3/19/25	28
M1 Data Cleaning	Gabriel	2/21/25	3/20/25	29
M1 Data Visualization	James	2/21/25	3/21/25	30
M1 Feature Reduction	Cliff	2/21/25	3/22/25	31
M1 Implementation & Coding	Tanishek	2/21/25	3/23/25	32
M1 Results Evaluation	All	2/21/25	3/24/25	33
Model 2 (M2) Design & Selection	Anirudh	2/21/25	3/25/25	34
M2 Data Cleaning	Gabriel	2/21/25	3/26/25	35
M2 Data Visualization	James	2/21/25	3/27/25	36
M2 Feature Reduction	cliff	2/21/25	3/28/25	37
M2 Coding & Implementation	Anirudh	2/21/25	3/29/25	38
M2 Results Evaluation	All	2/21/25	3/30/25	39
Midterm Report	Anirudh	2/21/25	3/31/25	40
Final Report				
Model 3 (M3) Design & Selection	Cliff	3/31/25	4/14/25	14
M3 Data Cleaning	gabriel	3/31/25	4/15/25	15
M3 Data Visualization	james	3/31/25	4/16/25	16
M3 Feature Reduction	tanishek	3/31/25	4/17/25	17
M3 Implementation & Coding	Anirudh	3/31/25	4/18/25	18
M3 Results Evaluation	Anirudh	3/31/25	4/19/25	19
M1-M3 Comparison	gabe	3/31/25	4/20/25	20
Video Creation & Recording	cliff	3/31/25	4/21/25	21
Final Report	All	3/31/25	4/22/25	22

PHASE THREE
Mar 14
Mar 21
Mar 28
Apr 4
Apr 11
Apr 18
Apr 25
W R F S U M T W R F S U M T W R F S U M T W R F S U M T W R F S U M T W R F S U M T W R F S U
Spring Break

Semiconductor Wafer Defect Analysis (WM811K)

Sample Wafer Image

This project analyzes semiconductor wafer defect patterns using the WM811K dataset, implementing various machine-learning techniques for defect classification.

Repository Structure

Key Directories

- Midterm.ipynb: Initial exploratory analysis and clustering

- Final.ipynb: Complete analysis with classification models

- Gantt charts (*.xlsx and *.png) tracking project timeline

- kmeans_clustering.png: Results from KMeans clustering
- PCA_RFT.png: PCA visualization for Random Forest
- confusion_matrix_RFT.png: Random Forest performance
- cnn_confusion.jpeg: Confusion matrix showing performance of the CNN model on classification.
- cnn_result.jpeg: Output predictions or visualizations generated by the CNN.
- cnn_training.jpeg: Accuracy/loss plot or training curve during CNN training process.
- dbscan_accuracy.jpeg: Evaluation metrics (accuracy or similar) for the DBSCAN clustering.
- dbscan_classify.jpeg: Classification result visualization after applying DBSCAN.
- dbscan_clusters.jpeg: Visualization of clusters discovered by DBSCAN.
- kmeans_accuracy.jpeg: Accuracy or performance evaluation after using KMeans.
- kmeans_classify.jpeg: Classification results or labeled data points using KMeans.
- kmeans_clusters.jpeg: Final clusters from KMeans plotted visually.
- forest_classify.jpeg: Intermediate or final clustering results from the Random Forest algorithm.
- forest_accuracy.jpeg: Accuracy or performance evaluation after using Random Forest.
- featureEngineering1.png: Step-by-step breakdown of the feature engineering process with grid visualization.
- featureEngineering2.png: Bar-plot visualizations showing distinct defect density patterns across failure types.
- pipeline.png: Overview of the full classification pipeline, from preprocessing to evaluation.
- defect_types.png: Visual reference or examples of the eight wafer defect types used in the dataset.
- predictions: Likely contains raw or formatted model predictions (file type unspecified).
- wafer.png: Sample wafer map used for analysis or visualization.
- wafer1.png: Another example or variation of a wafer map.
- wafer_ss.png: Screenshot or labeled example of a wafer used for sanity checking or showcasing labeling accuracy.

- hqdefault.jpg: Possibly a thumbnail image or external visual used for presentation or YouTube preview.

Key Files

- `README.md` : This documentation file
- `Proposal.pptx` : Initial project proposal slides

References

1. J. Wu, et al., “Wafer Map Failure Pattern Recognition and Similarity Ranking for Large-Scale Data Sets,” *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 1, pp. 102–109, 2015.
2. X. Fan, et al., “Wafer Defect Patterns Recognition Based on OPTICS and Multi-Label Classification,” *J. Intell. Manuf.*, vol. 27, no. 3, pp. 543–552, 2016.
3. J. Yu and X. Lu, “Wafer Map Defect Detection Using Joint Local and Nonlocal Linear Discriminant Analysis,” *IEEE Trans. Semicond. Manuf.*, vol. 28, no. 4, pp. 502–511, 2015.
4. H. Chen, et al., “Multi-Feature Fusion Perceptual Network for Wafer Defect Recognition,” *IEEE Access*, vol. 11, pp. 12345–12356, 2023.

Team Contributions

- Gabriel Feng: Wrote Intro/Background, Problem Definition, and References from proposal; Set up Jupyter Notebook environment; Implemented data preprocessing pipeline; Implemented and documented Random Forest classifier supervised method
- Anirudh Sriram: Analyzed model performance and interpreted confusion matrices; Wrote comprehensive methods and results sections, compiling a final report with integrated sections from team members
- Cliff Lin: Implemented and documented DBSCAN unsupervised method; Fine-tuned model parameters. Generated new confusion matrices (recall and precision) and classification accuracy bar graphs for presentation
- Tanishk Deo: Implemented and documented K-Means clustering unsupervised method; Created visualization and metrics of clustering results for KMEANS; Created CNN model for supervised learning and metrics for results;
- James Chen: Created visualizations of wafer defect patterns; Did all write-up and powerpoints presentaiton as well. Helped with visualizations Wrote final report