IESA® INDIA ELECTRONICS & SEMICONDUCTOR ASSOCIATION
India's ESDM Leadership and Knowledge Partner

NXP

GlobalFoundries™

Team Details

Team Idea ID: 667

Team Name: WAFER ENDEAVOURS

S.NO	ROLE	NAME	ACADEMIC YEAR
1	**Team Leader**	ROSHNI K	II YEAR
2	Member 1	REEVASRI S	II YEAR
3	Member 2	KANIMOZHI S	II YEAR
4	Member 3	SRRI LAKSHMI M R	III YEAR

COLLEGE NAME

Chennai Institute of Technology

☎ TEAM LEADER CONTACT NUMBER

+91 7358658891

✉ TEAM LEADER EMAIL ADDRESS

roshni.k2266@gmail.com

GitHub & Video Link

Phase 2 Submission

GitHub Repository

https://github.com/Roshni-K6/Edge-AI-Semiconductor-Defect-Classification.git

Prediction_log

https://drive.google.com/file/d/1qWFxBaPJs2gjT7vHXBHfT28SyZtR8rOf/view?usp=drivesdk

Stimulated_video_link

https://drive.google.com/file/d/1eCm32TkNfKMfPiUp0c3JhBi73sMIw_Rg/view?usp=drivesdk

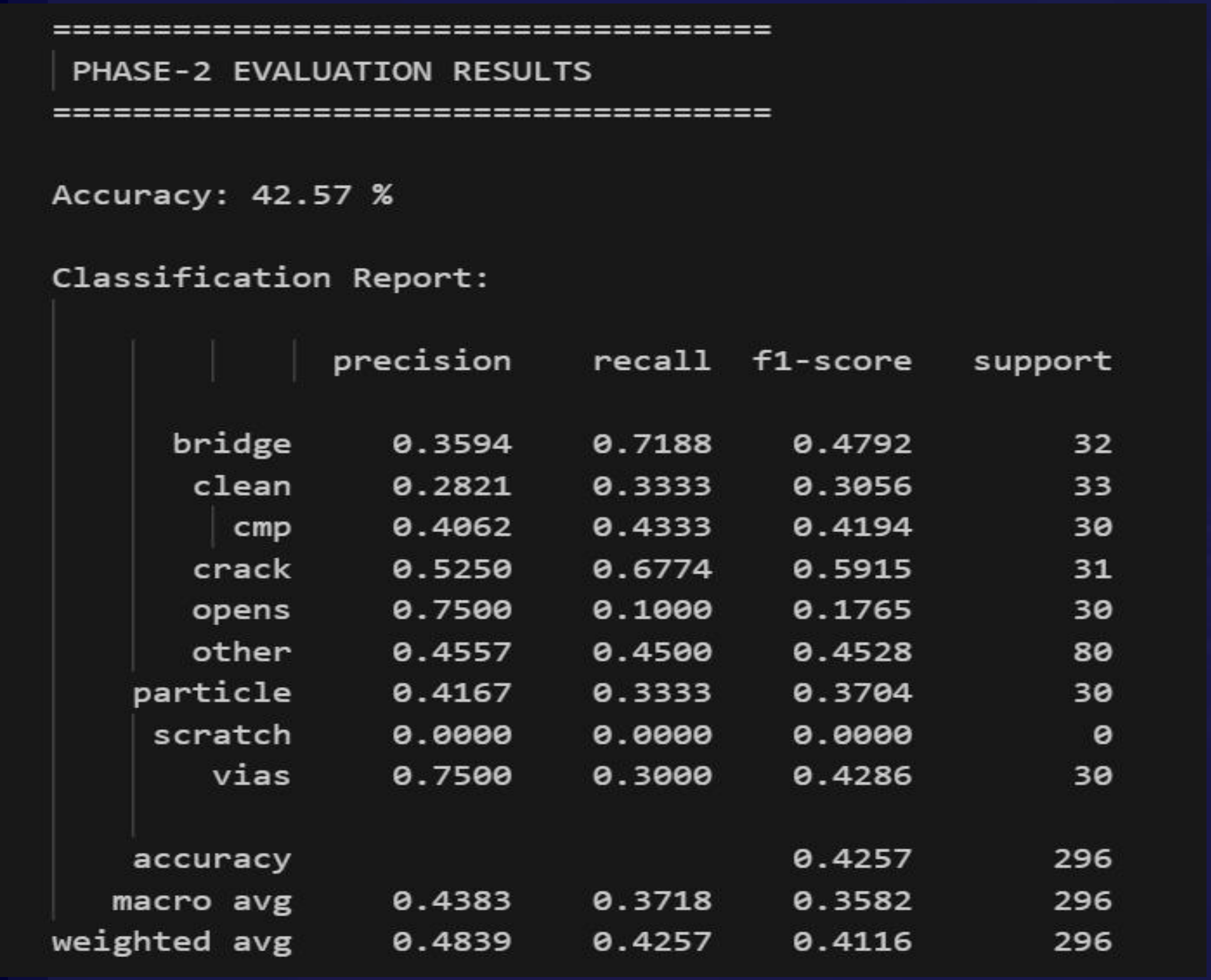
Wafer_Endeavours_Phase2

https://drive.google.com/file/d/1g4iVqF4BAtOy6PWQbEOhZHSjMCMU9_zz/view?usp=drivesdk

Prediction_code

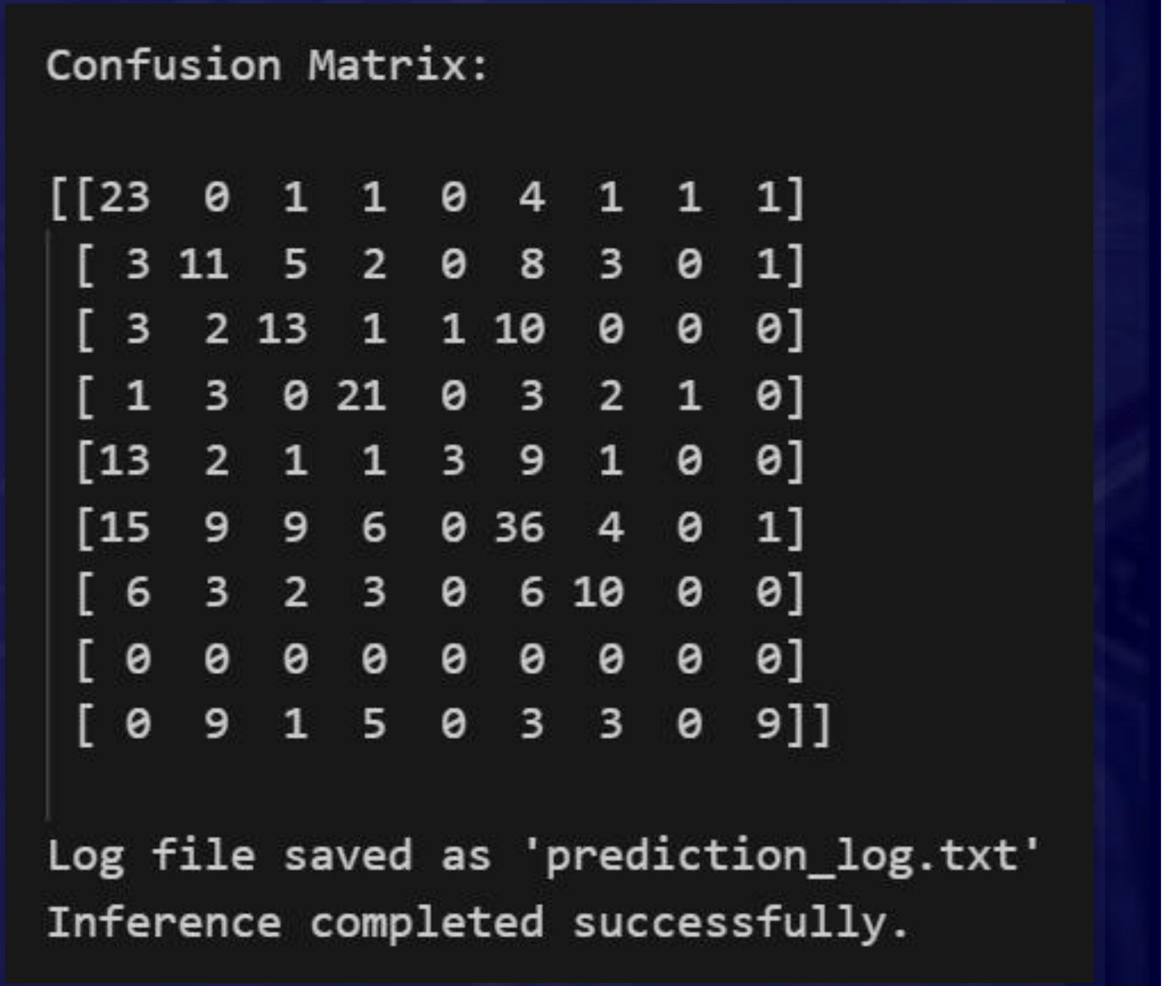
https://drive.google.com/file/d/1eT8MRZgVmR8YQyBUpIVi62Jzcw9krXO9/view?usp=drivesdk

CLASSIFICATION REPORT



```
====================================
 PHASE-2 EVALUATION RESULTS
====================================

Accuracy: 42.57 %

Classification Report:

              precision    recall  f1-score   support

      bridge     0.3594    0.7188    0.4792        32
       clean     0.2821    0.3333    0.3056        33
         cmp     0.4062    0.4333    0.4194        30
       crack     0.5250    0.6774    0.5915        31
       opens     0.7500    0.1000    0.1765        30
       other     0.4557    0.4500    0.4528        80
    particle     0.4167    0.3333    0.3704        30
     scratch     0.0000    0.0000    0.0000         0
        vias     0.7500    0.3000    0.4286        30

    accuracy                         0.4257       296
   macro avg     0.4383    0.3718    0.3582       296
weighted avg     0.4839    0.4257    0.4116       296
```

CONFUSION MATRIX



```
Confusion Matrix:

[[23  0  1  1  0  4  1  1  1]
 [ 3 11  5  2  0  8  3  0  1]
 [ 3  2 13  1  1 10  0  0  0]
 [ 1  3  0 21  0  3  2  1  0]
 [13  2  1  1  3  9  1  0  0]
 [15  9  9  6  0 36  4  0  1]
 [ 6  3  2  3  0  6 10  0  0]
 [ 0  0  0  0  0  0  0  0  0]
 [ 0  9  1  5  0  3  3  0  9]]

Log file saved as 'prediction_log.txt'
Inference completed successfully.
```

Problem Statement Addressed

Phase 1 Submission

Edge-AI Semiconductor Defect Classification

Semiconductor manufacturing generates large volumes of high-resolution wafer and die inspection images, where even microscopic defects can cause yield loss, device failure, and high rework costs.

Existing inspection pipelines rely on manual review or cloud-based AI, resulting in high latency and inability to support real-time defect detection.

Cloud-based processing incurs heavy data transfer and infrastructure costs, making large-scale deployment inefficient and expensive.

Delayed or inaccurate defect classification directly impacts production efficiency and overall yield in high-throughput fab environments.

Cloud dependency raises data privacy, security, and reliability concerns for sensitive semiconductor manufacturing data.

These challenges create a strong need for lightweight, accurate, low-latency, and energy-efficient Edge-AI defect classification systems that can operate on-site and at scale.

Idea Description

KEY CONCEPT AND APPROACH

The idea is to develop an Edge-AI based semiconductor defect classification system that performs real-time detection and classification of wafer and die defects directly on edge hardware.

The approach focuses on designing lightweight and compute-efficient AI models that balance accuracy, latency, and resource usage, reflecting real semiconductor fab constraints .

SOLUTION OVERVIEW

The solution ingests wafer and die inspection images and performs on-device AI-based defect classification into multiple predefined defect categories, including Clean and Other.

A custom dataset comprising clean and defective samples is used to train the model, ensuring balanced class representation and robustness across defect types.

By executing inference at the edge, the system eliminates cloud dependency, reduces latency and bandwidth usage, and preserves data privacy .

Proposed Solution

SOLUTION DETAILS

This project implements an edge-optimized image classification system to identify semiconductor wafer defects using a MobileNetV2 deep learning model. The solution is designed with edge deployment constraints in mind (low model size, fast inference) and exported to ONNX for compatibility with NXP eIQ / ONNX Runtime. The model classifies inspection images into multiple defect categories, along with Clean and Other, enabling automated quality control in semiconductor manufacturing.

DATASET PLAN & CLASS DESIGN

Total images (current): 1,200+ images

Number of classes: 9 classes (7 defect classes + Clean + Other)

Class list: clean, other, particle/contamination, scratch, opens, cracks, cmp, vias, bridges

Class balance plan: Minimum ~120 images per class to maintain balanced learning and avoid bias toward dominant defect types.

Train / Validation / Test split: 70% / 15% / 15%

Image type: Grayscale images (converted to 3-channel format for CNN compatibility)

Labeling method / source: Manually curated and labeled from publicly available semiconductor defect images sourced from research publications and open references. Data augmentation was applied to increase sample count while preserving defect characteristics.

Technology & Feasibility / Methodology Used

IMPLEMENTATION AND STRATERGY

A transfer-learning strategy is used where a pre-trained MobileNetV2 is adapted for semiconductor defect classification to reduce training time and data requirements. Images are standardized to a fixed input size and trained using a controlled train/validation/test split to ensure generalization. The trained model is optimized for edge deployment by exporting to ONNX, enabling low-latency inference on resource-constrained devices.

Software Architecture

Image preprocessing → CNN-based defect classification pipeline
MobileNetV2 with transfer learning for fast and efficient inference

Hardware Components

No dedicated hardware in Phase 1 (software-only solution)
Designed to run on standard CPU systems
Edge-ready for future deployment on NXP/embedded platforms

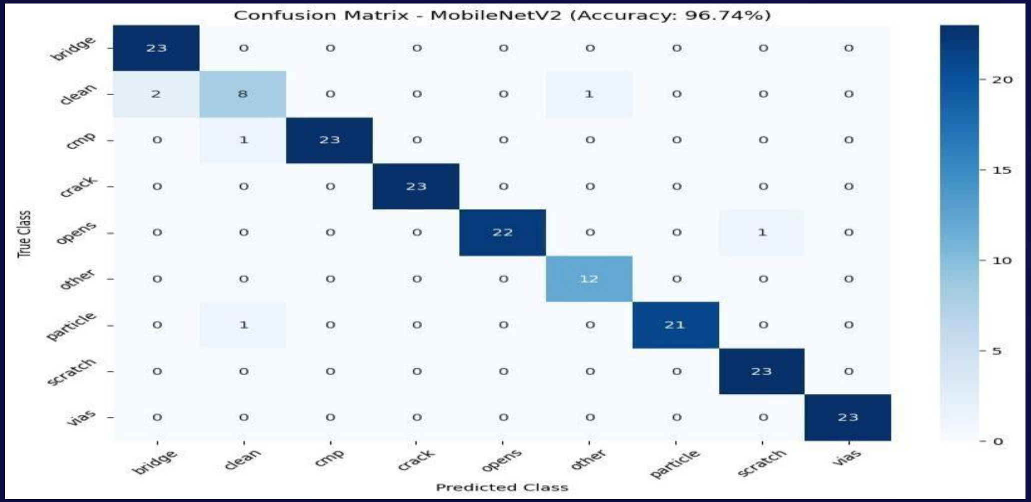
Development Tools

TensorFlow / PyTorch for model training
Python for preprocessing and evaluation
Google Colab + GitHub for development and version control

BASELINE MODEL AND RESULS

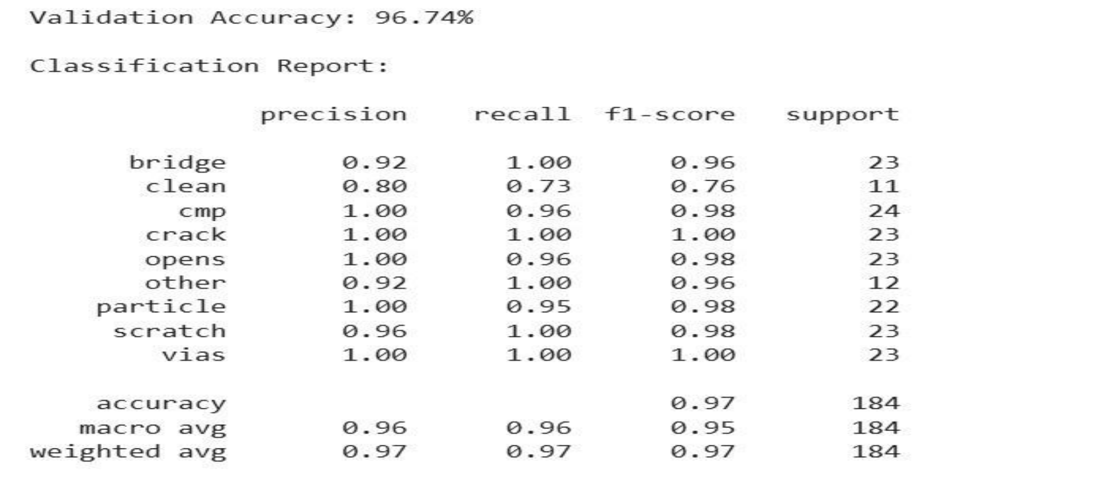
Architecture	MobileNetV2
Training Approach	Transfer Learning
Framework	PyTorch
Export Format	ONNX
Input Size	224 × 224 × 3
Model Size	~8.53 MB
Accuracy	96.74%
Precision / Recall	0.97 / 0.97

CONFUSION MATRIX



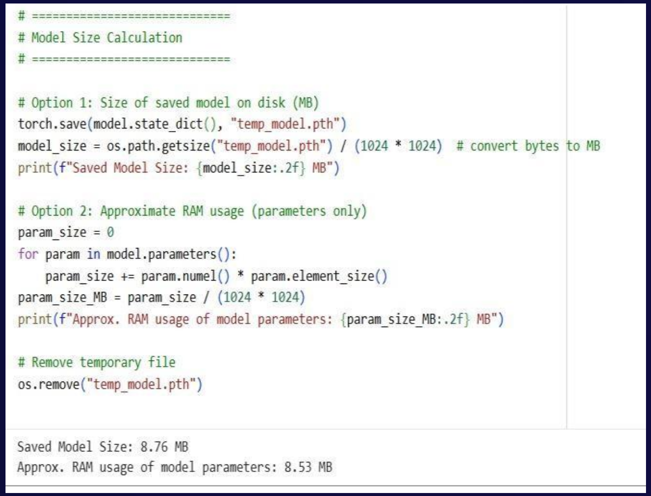
True Class \ Predicted Class	bridge	clean	cmp	crack	opens	other	particle	scratch	vias
bridge	23	0	0	0	0	0	0	0	0
clean	2	8	0	0	0	1	0	0	0
cmp	0	1	23	0	0	0	0	0	0
crack	0	0	0	23	0	0	0	0	0
opens	0	0	0	0	22	0	0	1	0
other	0	0	0	0	0	12	0	0	0
particle	0	1	0	0	0	0	21	0	0
scratch	0	0	0	0	0	0	0	23	0
vias	0	0	0	0	0	0	0	0	23

CLASSIFICATION REPORT



```
Validation Accuracy: 96.74%

Classification Report:

              precision    recall  f1-score   support

      bridge       0.92      1.00      0.96        23
       clean       0.80      0.73      0.76        11
         cmp       1.00      0.96      0.98        24
       crack       1.00      1.00      1.00        23
       opens       1.00      0.96      0.98        23
       other       0.92      1.00      0.96        12
    particle       1.00      0.95      0.98        22
     scratch       0.96      1.00      0.98        23
        vias       1.00      1.00      1.00        23

    accuracy                           0.97       184
   macro avg       0.96      0.96      0.95       184
weighted avg       0.97      0.97      0.97       184
```

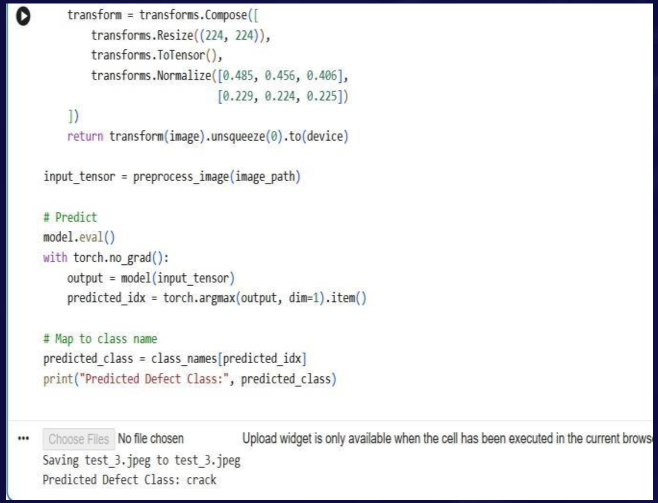
MODEL SIZE CALCULATION



```
# ==============================
# Model Size Calculation
# ==============================

# Option 1: Size of saved model on disk (MB)
torch.save(model.state_dict(), "temp_model.pth")
model_size = os.path.getsize("temp_model.pth") / (1024 * 1024)  # convert bytes to MB
print(f"Saved Model Size: {model_size:.2f} MB")

# Option 2: Approximate RAM usage (parameters only)
param_size = 0
for param in model.parameters():
    param_size += param.numel() * param.element_size()
param_size_MB = param_size / (1024 * 1024)
print(f"Approx. RAM usage of model parameters: {param_size_MB:.2f} MB")

# Remove temporary file
os.remove("temp_model.pth")
```

```
Saved Model Size: 8.76 MB
Approx. RAM usage of model parameters: 8.53 MB
```

PREDICTED DEFECT



```
    transform = transforms.Compose([
        transforms.Resize((224, 224)),
        transforms.ToTensor(),
        transforms.Normalize([0.485, 0.456, 0.406],
                             [0.229, 0.224, 0.225])
    ])
    return transform(image).unsqueeze(0).to(device)

input_tensor = preprocess_image(image_path)

# Predict
model.eval()
with torch.no_grad():
    output = model(input_tensor)
    predicted_idx = torch.argmax(output, dim=1).item()

# Map to class name
predicted_class = class_names[predicted_idx]
print("Predicted Defect Class:", predicted_class)
```

```
Upload widget is only available when the cell has been executed in the current brows
Saving test_3.jpeg to test_3.jpeg
Predicted Defect Class: crack
```

METRICS

ACCURACY : 96.74%

PRECISION / RECALL : 0.97

MODEL SIZE : 8.53 MB

GitHub & Video Link

GitHub Repository

https://github.com/Roshni-K6/Edge-AI-Semiconductor-Defect-Classification.git

Dataset ZIP Link

https://drive.google.com/file/d/1krg_vpDR0EoZHPNWtp0VPrj425JXW57d/view?usp=sharing

ONNX model Link

https://drive.google.com/file/d/12Pi88YtciSbqGKFJ_QCWeCkzd6X7Go-F/view?usp=drivesdk

Results report Link

https://drive.google.com/file/d/1QoalGWzkVoN7U2U4mKIoI5LcWjAGEJGd/view?usp=drivesdk