Journal of Intelligent Computing System (JICS), Vol.1, Issue. 3, pp. 01-20, (Sep 2026)
ISSN: 3139-3616 (online)

# An Integrated AI Automating Inspection Model for Surface Defect Segmentation and Classification in Construction Domain

Deepanshi Joon[1*], Aditi Nautiyal[2]
[1]Senior Engineer- AI, Tata Consulting Engineers
[2]Assistant Manager, Tata Consulting Engineers, Navi Mumbai, India
[1]abjoon90@gmail.com, [2]adi.nautiyal92@gmail.com

**Abstract-** Unlike old age, humans are no more bounded to live in mud houses. As they constructed strong modern setups like buildings, bridges and other infrastructures. But this too comes with some structural issues, which are cracks, stains, spalling and other flaws. If frequently, these structures are left untreated, then it may affect human safety and threaten serviceability. For inspection purposes, relying on manual visual examination will ultimately lead to its slow process. Moreover, this method is subjective, costly, error-prone and unsafe for surfaces which are inaccessible. Overall, these factors motivate automated computer vision inspection. However, there exists some constraints like limited, imbalanced datasets and inspection spans of various distinct types of tasks. Further, this work gives unified Deep Learning model which evaluates on one of the public benchmark datasets from Kaggle, which consists of sub folders of Magnetic-Tile Defect, DeepPCB, DBCC bridge cracks and CrackForest. In this paper multi-class classification, binary classification and pixel-level segmentation are considered. The model couple's residual backbone with classification, which is regularized head and symmetric U-Net. Overall, this model is trained from scratch without pre-training and deploys objective, which is imbalanced (class-weighted, positive weighted BCE with Dice, label-smoothed cross entropy). This framework attains F1 score of 0.9945 for DeepPCB dataset. The novelty lies in its approach, as its single, pre-training free pipeline which spans three task paradigms with the issue of handling imbalance and format agnostic ingestion and in real honest finding that scale of data, more than architecture and govern effectiveness. Further, Future work could focus on adding pre - trained backbones, multi-seed evaluation and board - disjoint and then end to end detection.

**Keywords:** *Surface defect detection, defect classification, semantic segmentation, convolutional neural network, from-scratch training, U-Net, class imbalance, computer-vision-based inspection, building inspection, crack detection, Deep Learning (DL).*

## 1. Introduction

Since its 2026, Human beings are surrounded by buildings, bridges, tunnels and other infrastructure, whether it's your home, shop, mall, paths or job place. Hence, this point is important to note down, that safety and cleanliness is prior fundamental need. Therefore, time to time inspection is required for their conditions that includes defect detections. Defects can be cracks, spalling, corrosion, peeling, staining and algae growth. These defects can't be neglected as this may lead to worst cases like catastrophic failure that endangers human lives. So, detection of these surfaces is not only related to defect detection rather a matter of public safety and for long term infrastructure sustainability. The same impact is related to manufactured components such as printed circuit boards, magnetic tiles and other parts related to industries [1]. Here, undetected surface defects directly give path to failures and safety concerns. Further, there comes hurdle which is - defect detection is still performed manually by trained personnel with their handled instruments. Here, the main issue is - its time consumption, cost, labour intensivist, it is more prone to some human error, require monitoring time to time and inter inspector variability. There too exists some difficult or dangerous places of surfaces. Which may be upper stories of high-rise buildings, undersides of bridges and confined industrial spaces [2]. Even somehow manually inspection done, then it is not going to be frequent neither completeness of inspection. These limitations pave the way for inspection methods which are safer, faster, more consistent and scalable.

This problem is then solved by using drones or other hardware to collect and click images. Then passing those images from Deep Learning Models. This process fasten ups the defect detection method. The limitations related to imbalance of class that dominates real defect data or frequently under addresses and methodology behaviour across storage formats, task types and class distributions that practical inspection presents. This paper addresses these gaps by presenting self-contained defect analysis pipeline and testing it across 4 heterogeneous Kaggle's public dataset. This pipeline consists of three task paradigms that are - binary crack detection, multi-class defect classification and pixel level crack segmentation.





*doi.org/10.67420/109319.1.3.1*

The model is trained from scratch, without any pretrained initialisation because its behaviour reflects the idea in isolation and so as it remains deployable when pre-trained weights are not available or restricted. This model is threefold – compact residual backbone with classification regularised head and structurally symmetric U-Net built from convolutional primitives which are same. Theses all three task paradigms under same training and inference protocol. Then, it also incorporates imbalance aware learning objectives overall, label-smoothed cross-entropy for classification and binary cross entropy positive weighted with Dice term for sparse foreground for segmentation task. This model also introduces robust, format agnostic data ingestion procedures which then includes box - guided reformulation that do convert detection bounding-box annotations into crop level classification enabling single model to consume four structurally datasets dissimilar without manual relabelling.

Next sections of this paper present background study, then comes section 3, which gives details for the proposed methodology (includes shared backbone, regularised classification head, segmentation network, dataset specific ingestion process, training objectives and optimisations). Further, Section 4 gives experimental results for four datasets as mentioned with cross dataset. End section gives outline for conclusion and gives direction for future work.

## 2. Background Study

This section deals with various review and research work for surface defect detections. Varying from year 2019 to 2026.

Y. Chen et al. [1] demonstrated a comprehensive review on surface defect detection models for various industrial products highlighting evolution from traditional ML techniques to DL approaches. Their paper categorised methods on various texture, shape and colour features. Their paper gave detailed overview about major challenges in industries for this, which includes real time processing, small defect localization, industrial defect detection and class imbalance. Further, their study used MVTec AD dataset and gave valuable insight into research work and future aspect in defect inspection for surface. D. Martin et al. [2] proposed adoption of practical DL for industrial defect detection. Their work identified key issues like high data acquisition costs, limited data availability, small size of defects, defect samples and the rarity related to defective instances. Based on these limitations, their framework gave effectiveness of DL models in construction industry system. Their work also validated industry case study for further providing and developing reliable automated defect detection solutions. Q. Jin and L. Chen [3] presented review work on surface defect models for limited amount of labelled data. The study further classified existing approaches into image processing traditionally. Gave a brief about conventional methods which then categorized in statistical, model learning approach and spectral, meanwhile DL models include augmentation, TL, fine tuning and other means. Their work presented that reducing large, annotated dataset dependences is important for industrial applications, as labelling and collection defect is often time consuming and labor intensive. Y. Ma *et al.* [4] reviewed systematic review of DL based techniques for object detection for surface defect industrial inspection. Their work analysed evolution of one stage, two stage supervised and unsupervised detection of defect algorithms. This highlighting their use cases in quality industrial inspection. This work discussed commonly utilised datasets and metrics evaluation meanwhile key challenges were identified. That includes variations in size of defects, trade-off among accuracy detection and speed processing and detection reliably of tiny defects. This review work also outlined further research directions focused at improving efficiency, robustness, and accuracy of DL defect detection models for real world environments in industrial manner. Y. Cheng *et al.* [5] gave comprehensive review of defect detection in industrial domain. This focuses on advances in Computer Vision (CV) and DL for real manufacturing use cases. This examined defect detection models across both modalities 2D and 3D with emphasis particularly on transition from traditional closed set ways to frameworks which were open set capable of focusing on previously defects which were unseen. This work further highlighted drawbacks which are practical like achieving automation, high precision, scalability and reliance reduced on defect extensive annotations. Their work did analysis for emerging trends in research and opportunities for future. Which offers perspective comprehensive on development of intelligent industrial detection of defect systems. X. Wang *et al.* [6] showcased review of ML based surface detection defection model for products in industrial manner. Which covers traditional processing, ML and DL criteria. The work proposed classification structured framework, limitations, strengths and then industrial applications were analysed for each type. This review also summarized defect datasets and then metrics were evaluated for recent advancements in large DL models for detection of defect. Additionally, it focused on practical aspects like hardware configuration, data preparation, model deployment and evaluation of performance. Meanwhile, key challenges identifying were related to quality of data, accuracy of detection.

G. Qin *et al.* [7] analysed recent advances in DL for surface defect detection of metals, with focus on manufacturing of steel applications. Their work examined enhancements of methods of supervised learning and then discussed

emerging methods like defect generation of data, unsupervised and semi-supervised learning anomaly detection to deal with challenges of limited data (labelled). The work further also explored growing role of fundamental models in transforming detection of defect paradigms. This study focuses, their potential for adaptability improving and generalization. Next to it, this work discussed about integration of AI in inspection systems of industries and then emphasized future opportunities for achieving autonomous intelligent and quality control closed loop in industries. L. Zhao *et al.* [8] gave survey systematic on ML vision-based defect surface detection. Then focusing on main limitation encountered in inspection of industries. This involves complex backgrounds, small defect detection, class imbalance, dynamic scenes and then generalization for cross-domain. The work reviewed recent methods made to tackle these problems and conducted analysis of more than 40 publicly datasets (available) industrial spanning diverse applications like photovoltaic panels, metal surfaces, printed circuit boards and pavements. Moreover, this work gave evaluation metrics on quantitative dataset and focused on emerging role of foundation models, which were large scale and benchmark datasets (high quality) in enhancing scalability, robustness, and practical deployment of defect detection systems in industries. Y. Cheng *et al.* [9] reviewed comprehensive work for industrial surface defect detection and analysed defect detection models with emphasis particular on transition from closed set to the direction of open set works capable of previously identifying defects which were unseen. The work discussed advantages of open set methods in reducing dependence on defect annotations extensively. F. Shang *et al.* [10] analysed recent DL techniques for steel surface defect detection (strip) focusing on advancements among 2020 and 2025. The authors focus widespread adoption of object (single-stage) detectors specifically YOLO-based techniques for inspection, meanwhile their limitations in detecting tiny low-contrast defects. Work discussed advanced changes like attention mechanisms, multi-scale feature fusion and strategies improvement in data augmentation to advance performance of detection. Adding up, it focused on importance of multimodal sensor fusion, self-supervised learning and digital twin driven quality control as directions for tackling computational constraints, data scarcity and challenges deployment in manufacturing surrounding.

Q. Qin et al. [11] gave Hierarchical Depth-Aware Refinement YOLO (HDR-YOLO) lightweight object detection model for real-time and efficient defect metal surface detection under industrial complex conditions. This approach showcased Query-Focused Convolution (QFC) module to give texture high-resolution and edge features in shallow layers. Meanwhile, applying Query-Based Fusion (QBF) module in layers, which were deeper to strengthen global semantic representation by adaptive interaction feature. By assigning different refinement approaches to various network depths model balanced feature effectively representation and efficiency computationally. Experimental evaluations on GC10-DET and NEU-DET datasets demonstrated improvements of 7.67% and 3.92% in mAP@0.5, respectively. And this over baseline model meanwhile competitively maintaining inference speed. Their work at end added that refinement in hierarchical depth-aware feature is strategy effective for enhancing lightweight defect detection performance in industrial environments manufacturing. E. Somtochukwu and N. Rijal [12] proposed edge-optimized framework which is Industrial-YOLO. Their work was based on fine-tuned YOLOv8 model for detection of industrial surface defect. Their work addressed limitation of deploying DL models on constrained of resource edge devices meanwhile handling high accuracy of detection and inference low latency. The model was evaluated utilising NEU on surface defect dataset. The other dataset, and additional automotive data of manufacturing consisting of structural defects like inclusions, pits and scratches. To improve efficiency for deployment, work integrated OpenVINO and TensorRT techniques optimization. This enabled accelerated inference on hardware edge. Outcomes of their work demonstrates speed inference exceeding FPS 120 on NVIDIA Orin Jetson platform meanwhile attaining mean Average Precision (mAP) value as 98.5%. Their model gave approach which presented real-time performance in automatically environments assembly line. This highlighted their potential for efficient and scalable Automated Optical Inspection (AOI) work in modern world manufacturing. P. Bergmann et al. [13] developed model on MVTec Anomaly Detection (MVTec AD) dataset for unsupervised industrial anomaly detection. This dataset of 5,354 sample size of high resolution spanning multiple texture and object categories, with images which were defect-free provided for anomalous and training images reserved for testing purpose. Their work includes more than 70 kinds of surface defects like scratches, contaminations, dents, and irregularities in structural along with pixel-level ground truth annotations for localization accurately. This work conducted summarised performance of already existing anomaly unsupervised detection methods. This work involved autoencoders convolutional, GAN, feature methods for descriptor-based and classical CV methods. Results showcased good scope for enhancing performance anomaly detection. This established MVTec AD as standard benchmark dataset for evaluating algorithm for industrial surface defect detection.

K. Roth et al. [14] presented PatchCore, which is detection of unsupervised anomaly model architected to tackle cold start issues in inspection time of industries via training on defect-free images exclusively. Models construct maximally

showcasing memory bank of nominal features extracted on patch level from pretrained CNN assuring accurate anomaly localization and detection without required training for defective samples. Integrating feature embeddings with nearest-neighbour efficient search strategy, PatchCore attained both high accuracy for detection and inference speed. Evaluations experimental on MVTec AD shows state of art performance. This achieved anomaly detection on image level. Higher AUROC shows outperforming output of existing work. This model also showcased strong capabilities among additional datasets and in few learning shot scenarios making the solution practical for industrial detection of anomaly work. M. Rudolph et al. [15] proposed network of Asymmetric Student Teacher (AST) for detection of industrial anomaly utilising only defect free samples for training. Not like, conventional student teacher architectures. The model's architecture adds up normalizing flow as of teacher and feed forward model as student. This improves distinction among anomalous samples. Results on MVTec 3D-AD and MVTec AD datasets showcases strong performance for both 3D anomaly and RGB detection. P. Mishra et al. [16] developed DRAEM that was discriminatively trained for model of reconstruction embedding for surface defect detection. The architecture sum-ups image reconstruction with learning (discriminative) to integrally analyse images which are anomalous and their reconstructed counterparts. This enabling anomaly accurate localization without adding up post processing. Evaluations on DAGM and MVTec AD datasets showed that DRAEM performed more well than existing methods that were unsupervised and got performance comparable to supervised approaches meanwhile providing accuracy (superior localization).

T. Defard et al. [17] developed PaDiM which is distribution patch modelling model for anomaly detection in one class learning setting. Their method used CNN features (pretrained) and multivariate Gaussian distributions to model normal data meanwhile exploiting correlations feature around multiple levels of semantic for enhanced localization. Outcomes on STC and MVTec AD datasets showed that PaDiM performed better than existing models in both anomaly localization and detection. Further, made it well suited for industrial practical inspection roles. M. Salehi et al. [18] developed multiresolution distillation architecture for anomaly detection. By transferring feature representations (from pretrained expert network) to lightweight student network. The work detects anomalies based on various discrepancies among intermediate feature showcasing of two networks and employs techniques for precise localization of anomaly without actual requiring complex procedures of training. Outcomes on MVTec AD dataset and other benchmark datasets showed competitive or performance superiority in both anomaly localization and detection compared with already existing methods. V. Zavrtanik et al. [19] developed DSR which is dual subspace network re-projection for surface unsupervised anomaly detection which generates anomalies (in feature space) instead of depending on anomalies of synthetic image level. The model employs quantized feature. This model's dual decoders to enhance detection of near in distribution anomalies without any assumptions about their characteristics visual. Their work's results on MVTec AD and KSDD2 datasets showed great performance, attaining good improvements over existing unsupervised methods. X. Tao et al. [20] analysed recent works in DL based unsupervised anomaly localization images (industrial based). The survey showed existing work based on their learning strategies. This summarized commonly utilised benchmark datasets and did comparativeness with their performance. Their work also identified main challenges including high annotation costs, limited defect samples and deployment in manufacturing environments. Meanwhile outlining future research opportunities.

Further, Section 3 and 4 will demonstrate how this work is reliable source for surface defect detection.

## 3. Proposed Methodology

This section will give a walkthrough of proposed models parts for this study, it consists of overview, mathematical aspects, formulations, Segmentation part, augmentation, architectural network and algorithms related to this proposed model. Next to this section, proposed model's results section presents defect detection outputs. Overall, this study concludes with conclusion and references at end of this study.

### 3.1 Overview and Notation

This study presented UDefNet (which is Unified Defect Network) (as shown in Figure 1), a single self-contained DL for automating surface defect detection which is instantiated among four different industrial benchmarks and 3 task paradigms (as shown in Table I) -

**Table I.** Descriptions of dataset subsets and various related classes

| Dataset | Task paradigm | Output | Classes |
|---|---|---|---|
| Magnetic-Tile-Defect (MTD) | Multi-class classification | image-level label | 5 (blowhole, break, crack, fray, free) |

| | | | |
|---|---|---|---|
| DBCC Bridge Crack | Binary classification | image-level label | 2 (crack, no-crack) |
| DeepPCB | Multi-class classification (detection→ crop) | region-level label | 6 (open, short, mouse bite, spur, copper, pinhole) |
| CrackForest (CFD) | Semantic segmentation | per-pixel mask | 2 (crack, background) |

This model was trained from scratch with no external pretraining or ImageNet, so as it is fully deployable and reproducible at places where pretrained weights are restricted or unavailable. It is single residual convolutional backbone. Segmentation task was served by three classifications tasks, symmetric structurally U-Net was built from same primitive convolution. Common augmentation policy was shared for all four categories (somewhat imbalance aware objective family). Warmup-cosine optimization plus AdamW were scheduled under automation of mixed precision, and TTA- test time augmentation inference rule.

**Notation:** Let $x$ denote input image and further, let it be like - $x \in \mathbb{R}^{3\times H\times W}$ and $y \in \{1, \dots K\}$ (here, $y$ is its label) for classification, $m \in \{0,1\}^{H\times W}$(which is binary mask for segmentation). $Conv_k^s$ denotes $k \times k$ convolution (stride omitted when 1) with stride $s$, Layer Normalization as LN, Global Average Pooling as GAP, Batch Normalization as BN. $f_\theta$ denotes network (full) with parameters $\theta$, softmax$(f_\theta(x))$ = $\hat{p}$ (which is predicted class distribution).

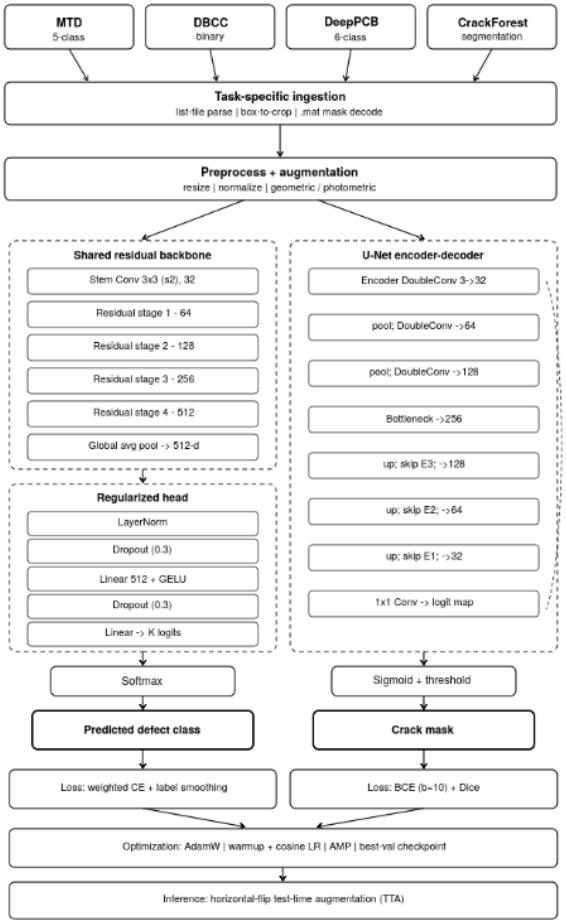


**Figure 1.** Proposed architectural flow for surface defect detection

### 3.1.1 Formal problem statements

This subsection gives problem statements in formulative manner- for classification, segmentation, and detection to classification.

**Classification** - Given $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ labeled corpus, learn the probability simplex ($f_\theta: \mathbb{R}^{3\times H\times W} \to \Delta^{K-1}$) minimizing the classification risk, which was expected, evaluated by per-class recall, precision, macro F1and confusion matrix.

**Segmentation** - Given, $\mathcal{D} = \{(x_i, m_i)\}_{i=1}^{N}$, learn producing crack probability per pixel $g_\theta: \mathbb{R}^{3\times H\times W} \to [0,1]^{H\times W}$, evaluated by IoU which is Intersection over Union, pixel level precision/F1/ recall and Dice coefficient.

**Detection to Classification (DeepPCB)** - Given annotation set $\{(b_j, c_j)\}$ for board images of boxes $b_j$ and defect codes $c_j$, rebuilt gained classification corpus $\mathcal{D}' = \{(crop\,(x, b_j), c_j)\}$ and further solve it as task of classification, therefore decoupling defects detection from localization.

### 3.2 Shared Residual Backbone

The backbone of classification $\phi_\theta$ maps image (input) to 512-dimensional embedding via stride stem, 4 residual phases (2 blocks each) and then global average pooling. Following Table II summarizes the full architecture for input $224 \times 224$.

*Table II. Backbone architecture (input 3×224×224).*

| Stage | Layer(s) | Stride | Params (approx.) | Output ($C \times H \times W$) |
|---|---|---|---|---|
| Stem | $\mathrm{Conv}_3$–BN–ReLU | 2 | ~0.9 K | $32 \times 112 \times 112$ |
| Stage 1 | ResBlock $32 \to 64$; ResBlock $64 \to 64$ | 2, 1 | ~0.13 M | $64 \times 56 \times 56$ |
| Stage 2 | ResBlock $64 \to 128$; ResBlock $128 \to 128$ | 2, 1 | ~0.52 M | $128 \times 28 \times 28$ |
| Stage 3 | ResBlock $128 \to 256$; ResBlock $256 \to 256$ | 2, 1 | ~2.1 M | $256 \times 14 \times 14$ |
| Stage 4 | ResBlock $256 \to 512$; ResBlock $512 \to 512$ | 2, 1 | ~8.4 M | $512 \times 7 \times 7$ |
| Pool | Global average pool + flatten | – | – | 512 |

The budget of parameter is heavily focused in Stage 4 which is (approx. 74% of the aprox.11.4 M total), showcasing quadratic growth of convolutional parameters (with channel width). Next to it, for $128 \times 128$ input (utilized for the small PCB and crack crops), $128 \to 64 \to 32 \to 16 \to 8 \to 4 \to 1$ is spatial sequence, the parameter count is same-means unchanged because convolutions are independent of resolution.

#### 3.2.1 Residual block

Each block of residual computes-

$$y = ReLU\big(F(x) + \mathcal{S}(x)\big),\ F(x) = BN\left(Conv_3\left(ReLU\big(BN(Conv_3^s)\big)\right)\right) \quad \text{Eq. (1)}$$

with shortcut of identity $\mathcal{S}(x) = x$ when output and input shapes match and shortcut of projection $\mathcal{S}(x) = \mathrm{BN}\big(\mathrm{Conv}_1^s(x)\big)$ (as shown in Eq. 1) when the channel stride or count changes. Further, two design consequences are up to mark. First, residual formulation gives gradient highway that keeps early gradients of layer well organized, which is critical for from basic convergence on datasets of few hundred to tens of thousands of images. Next, stacking $3 \times 3$ convolutions present growing effective receptive field, after stem and then 4 stages theoretical receptive field exceeds input extent. 512-d embedding integrates global image context enabling recognition of spatially defects (example - peeling region) also localized ones (example a blowhole).

### 3.3 Regularized Head of Classification

The $z = \phi_\theta(x) \epsilon \mathbb{R}^{512}$ classification is classified by 2-layer head along with normalization and stochastic regularization rather than single project which is linear projection-

$$h(z) = W_2\left(Drop_p\big(GELU(W_1 Drop_p(LN(z))\big)\right) + b_2 \quad \text{Eq. (2)}$$

with $W_1 \in \mathbb{R}^{512\times512}, W_2 \in \mathbb{R}^{512\times K}$ (as shown in Eq. (2)) and dropout rate as $p = 0.3$. Components serve different purposes- LN standardizes embedding segregations entering classifier and then decreasing sensitivity to feature-scale

drift among mini-batches. Intermediate GELU projection enhances capabilities of decision function far from linear boundary and 2 dropout layers act as implicit gathers that curbs overfit on classes with some number of samples. Head adds approx. 0.26 M parameters in it. Replacing this head with bare linear layer is bare minimum expected to reduce generalization on most imbalanced and smallest corpora, which motivates its involvement.

**Initialization:** Convolutional kernels utilize initialization of Kaiming-normal that is ReLU gain and fan_out. Linear weights utilize truncated-normal that is $\sigma = 0.02$ with zero bias. BN affine parameters are parameterized to unit zero or scale shift. This idea is chosen particularly from-scratch optimization for stability.

### 3.4 Network for Segmentation

For Crack Forest part, Fundamental backbone is replaced by symmetric (U-Net) encoder–decoder which reuses similar convolutional basics. Further, Table III mentions topology for input $256 \times 256$.

**Table III.** Architecture of U-Net (input as 3×256×256).

| Path | Block | Output (C × H × W) |
|---|---|---|
| Encoder E1 | DoubleConv $3 \rightarrow 32$ | $32 \times 256 \times 256$ |
| Encoder E2 | MaxPool; DoubleConv $32 \rightarrow 64$ | $64 \times 128 \times 128$ |
| Encoder E3 | MaxPool; DoubleConv $64 \rightarrow 128$ | $128 \times 64 \times 64$ |
| Bottleneck E4 | MaxPool; DoubleConv $128 \rightarrow 256$ | $256 \times 32 \times 32$ |
| Decoder D3 | UpConv $256 \rightarrow 128$; concat E3; DoubleConv | $128 \times 64 \times 64$ |
| Decoder D2 | UpConv $128 \rightarrow 64$; concat E2; DoubleConv | $64 \times 128 \times 128$ |
| Decoder D1 | UpConv $64 \rightarrow 32$; concat E1; DoubleConv | $32 \times 256 \times 256$ |
| Head | $\mathrm{Conv}_1 \rightarrow 1$ | $1 \times 256 \times 256$ |

Each *UpConv* is $2 \times 2$ transposed convolution and each *DoubleConv* is $[\mathrm{Conv}_3$–BN–ReLU$] \times 2$. Further, skip connections add up encoder feature maps into decoder. This reinstates high frequency detail of spatial, which on other hand affected by pooling. This is important for delineating one to few pixel wide and thin crack structures in CFD phase. The single-channel map of logit is passed via sigmoid at inference to give probabilities of per-pixel crack.

### 3.5 Specific Dataset Ingestion

Different aspects of model are set of robust ingestion processes, which map every benchmark's idiosyncratic data format onto the common interface of training interface, which too is without manual relabeling.

- **DBCC-** In list files Labels reside (train.txt, val.txt) with binary code pair filename. Algorithm 1 resolves each entry to physical file via means indexing of filename, tolerant of arbitrary placement of sub-directory and saves dataset's native partition of train/validation. Validation list is stratified split into test subsets and disjoint validation.

- **MTD-** Images are collected by traversal recursive of per-class directory structure like tree, retaining only extensions which are photographic and excluding explicitly annotation/mask files which share class folders. Because masks are never mistaken for samples of input.

- **CrackForest-** Decoded masks from MATLAB .mat files, label of background is inferred as value modal of each map of segmentation and all other labels are combined in crack foreground, mask extraction making invariant to label-encoding conventions.

- **DeepPCB (Algorithm 2)-** Specifically for each defective board image and its related annotation file, every box of bounding is expanded by fixed padding margin $\rho$ to retain context of local (resized, cropped, and stored under its defect-type class). Crops union forms derived corpus $\mathcal{D}'$ classification.

**Algorithm 1: label-list ingestion for DBCC**
**Input:** L list file, R image root
**Output:** S as sample set = {(path, label)}
Step 1: {basename(f): fullpath(f) in walk(R) if f is image} to index
Step 2: "Ø "to "S"
Step 3: in L, for each line:

Step 4: tokens ← split(line), lab ← integer token, name ← remaining token
Step 5: S ∪ {(index[basename(name)], lab)}: if basename(name) in index to S
Step 6: return S

**Algorithm 2: classification cropping for DeepPCB detection**
**Input:** "X" board images, "ρ" padding , "A" annotation files
Output: {(crop, class)}= crop corpus D′
Step 1: ∅ to D′
Step 2: for every defective image x with "a" as annotation:
Step 3: for each (x1,y1,x2,y2,t) in a:
Step 4: clip([x1−ρ, y1−ρ, x2+ρ, y2+ρ], image bounds) to b
Step 5: if b is valid box:
Step 6: D′ ∪ {(crop(x, b), name(t))} to D′
Step 7: return D′

### 3.6 Augmentation and Data Pipeline

To fixed square resolution, inputs are resized per task and then normalized with channel statistics, which are standard ($\sigma = [0.229,0.224,0.225], \mu = [0.485,0.456,0.406]$) (Table IV briefs augmentation settings).

**Table IV.** Preprocessing and Augmentation

| Setting | Classification (train) | Evaluation | Segmentation (train) |
|---|---|---|---|
| Resolution | $224^2$ (MTD) / $128^2$ (PCB, DBCC) | task resolution | $256^2$ |
| Geometric | random-resized-crop, H/V flip, rotation $\pm(15–20)°$ | resize + center-crop | H/V flip, 90° rotation (applied jointly to image+mask) |
| Photometric | brightness/contrast/saturation/hue jitter | none | brightness/contrast (image only) |
| Occlusion | random erasing ($p = 0.2–0.25$) | none | – |

Further, Geometric transforms are implemented identically to mask and image to preserve pixel correspondence for segmentation, only photometric jitter is applied to image. Classification corporation utilizes stratified 60/20/20 train/validation/test split (DBCC uses its native split) which guarantees proportional representation of class.

### 3.7 Objectives of Training

#### 3.7.1 Imbalance-aware loss of classification

With $N$ samples on $K$ classes and $n_c$ samples in $c$ class, the inverse-frequency class weight is defined as $w_c = N/(K\,n_c)$ (as shown in Eq. (3)). Combined with "ε" label smoothing, Following is objective –

$$\tilde{y}_c = (1-\varepsilon)\,y_c + \varepsilon/K\,,\ \mathcal{L}_{\text{cls}} = -\sum_{c=1}^{K} w_c\ \tilde{y}_c \log \hat{p}_c \quad \text{Eq. (3)}$$

Weighing of Inverse-frequency safeguards majority class (exampled, approx. 7:1 ratio no-crack: crack in DBCC, or in MTD - dominant defect-free class) from dominating gradient. Label smoothing regularizes logit magnitudes and then improves calibration of probability.

#### 3.7.2 Compound - segmentation loss

Pixels of crack form small fractions of each image. To combine binary positive-weighted cross-entropy with term-Dice overlap. Further, with ground truth $m_i$, $\hat{m}_i = \sigma(\text{logit}_i)$, positive weight $\beta = 10$, pixel set $\Omega$, and smoothing $\epsilon$-

$$\boldsymbol{\mathcal{L}_{\text{BCE}}} = -\frac{1}{|\Omega|}\sum_{i\in\Omega}\left[\beta\, m_i \log\hat{m}_i + (1-m_i)\log(1-\hat{m}_i)\right],\ \boldsymbol{\mathcal{L}_{\text{Dice}}} = 1 - \frac{2\sum_i \hat{m}_i m_i + \epsilon}{\sum_i \hat{m}_i + \sum_i m_i + \epsilon},\ \boldsymbol{\mathcal{L}_{\text{seg}}} = \mathcal{L}_{\text{BCE}} + \mathcal{L}_{\text{Dice}}. \quad \text{Eq. (4)}$$

Here, term BCE gives stable, dense per-pixel gradients meanwhile its positive weight counters scarcity of foreground, Dice term (as shown in Eq. (4)) directly maximizes overlap region, and it is scale-invariant to fraction foreground, jointly avoiding degeneration of all-background solution.

### 3.8 Optimization

This subsection briefs about hyperparameters and algorithms for this study (as shown in Table V), followed by other subsections and proposed results section for this study's problem statement.

**Table V.** Optimization hyperparameters.

| Hyperparameter | Classification | Segmentation |
|---|---|---|
| Optimizer | AdamW | AdamW |
| Base LR $\eta_0$ | $10^{-3}$ | $10^{-3}$ |
| Weight decay | $10^{-4}$ | $10^{-4}$ |
| Warmup epochs | 3–4 | 5 |
| Schedule | warmup + cosine | warmup + cosine |
| Label smoothing $\varepsilon$ | 0.05–0.1 | – |
| Dropout $p$ | 0.3 | – |
| Precision | mixed (AMP) | mixed (AMP) |
| Model selection | best val macro-F1 | best val IoU |

The linear warmup is followed by learning rate then cosine annealing-

$$\eta(t) = \begin{cases} \eta_0 \, \frac{t}{T_w}, & t < T_w, \\ \frac{\eta_0}{2}\left(1 + \cos(\pi \, t - T_w/T - T_w)\right), & t \geq T_w, \end{cases} \qquad \text{Eq. (5)}$$

With $T_w$ as warmup steps, $t$ as step index, $T$ as total steps (as shown in Eq. (5)). cosine annealing enables fine convergence, and warmup stabilizes high-variance at early phase of from-start. As there exists no pre-trained weights, single uniform base rate of learning is implemented to all parameters, which contrasts with schedules of discriminative learning-rate used in fine-tuning. Precision, which is mixed decreases training of accelerates and memory. Further, they used gradient scheduler and scaleris advanced only on steps that update optimizer. As skipped, overflow-detected steps do not lead to advanced scheduling.

**Algorithm 3: From-initial training with warmup-cosine and AMP**
**Input:** model val loader D_val, epochs E
f_θ, loader D_train,

Step 1: best ← −∞
Step 2: for epoch = 1..E:
Step 3: for (x,y) in D_train-
Step 4: with autocast: ℓ ← Loss(f_θ(x), y)
Step 5: scaler.scale(ℓ).backward()
Step 6: s0 ← scaler.scale; scaler.step(opt); scaler.update()
Step 7: if scaler.scale ≥ s0: scheduler.step()
Step 8: v ← Validate(f_θ, D_val)
Step 9: if v > best: best ← v; save(θ)
Step 10: return argmax-val checkpoint

### 3.9 Test-Time Augmentation involving Inference

At time of testing predictions are stabilized via horizontal flip TTA that averages evidence from that of mirrored and original views before decision-

$$\text{classification} \qquad \hat{y} = \operatorname{argmax}\left(f_\theta(x) + f_\theta\left(\text{flip}(x)\right)\right) \qquad \text{Eq. (6)}$$

$$\text{segmentation} \qquad \hat{m} = \mathbb{1}\left[1/2\,(\sigma\left(g_\theta(x)\right) + \sigma\left(\text{flip}\left(g_\theta\left(\text{flip}(x)\right)\right)\right)) > \tau\right] \qquad \text{Eq. (7)}$$

Here $\tau$ is selectable decision threshold on set of validation. Crack segmentation benefits from $\tau$ tuned below 0.5 owing to scarcity of foreground (as shown in Eq. (6) and Eq. (7)). Averaging of symmetric decreases prediction variance at cost of negligible and gives small consistent and metric enhancement.

### 3.10 Parameter Budget and Computational Complexity

The backbone of classification consists of approx. 11.4 M parameters dominated by widest stage of 512 channel. This is on average order of smaller magnitude than same backbones which are pre-trained example VGG-16 at approx. 138

M, ResNet-50 at approx. 25 M giving inference cost and lower memory. This is advantageous for edge deployment, especially in inspection field for drones and mobile robots. As for these models from scratch are trained. At any stage of pipeline there is no such dependency on external level. Cost convolutional scales as $O(\sum_l C_{l-1} C_l k^2 H_l W_l)$ per-stage down-sampling and stride stem reduce geometrically $H_l W_l$. While keeping Stage-4 dominant cost tractable. The cost U-Net's is dominated via its encoder/decoder ends in full-resolution. This segmentation consistent with finer fidelity spatial required.

### 3.11 Contribution's Summary

1. **UDefNet framework** tackling multi-class classification, semantic segmentation and binary classification with primitives shared convolutional and training/inference common protocol which is free of pre-trained dependencies.
2. **Regularized multilayer classification head** enhancing generalization on imbalanced small defect corpora on linear classifier.
3. **Objectives of imbalance-aware**: Label-smoothed cross-entropy, inverse-frequency-weighted for positive-weighted BCE + Dice compound loss and classification for sparse-foreground segmentation of crack.
4. **Format-agnostic and robust ingestion**- involving box-guided detection to classification reformulation for DeepPCB and modal background mask CrackForest decoding, ensuring single framework to get four dissimilar structural output results.
5. **Experimental Results and Discussion**

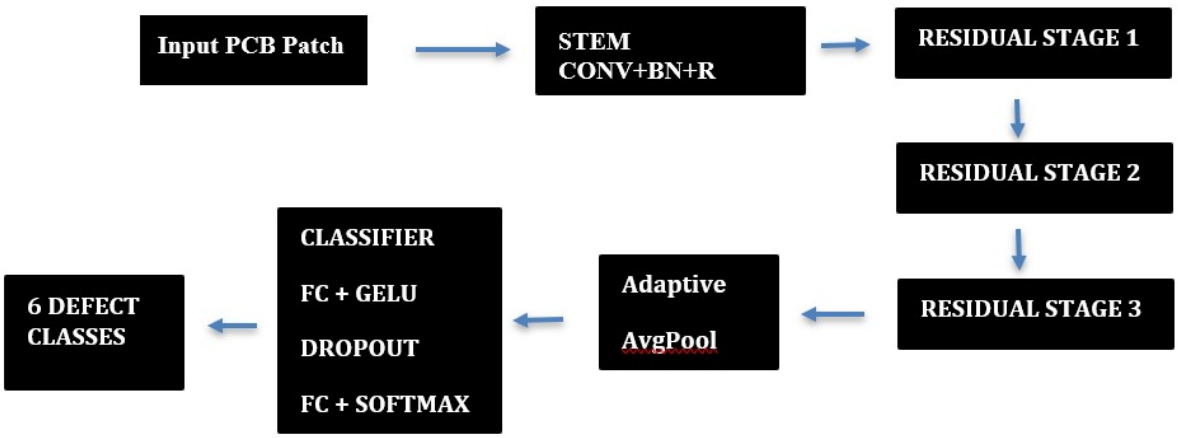


**Figure 2.** Architectural flow for Shared Residual Backbone

The presented **DefectNet** model is CNN based (as shown in Figure 1), compact especially designed for defect type classification in PCB that is Printed Circuit Board imagery (as given shared residual backbone model in Figure 2). The architecture works on defected cropped patches of $128 \times 128$ pixels size and then trained from zero situation to learn discriminative spatial relation patterns for 6 defect parts. Which are open, mouse bite, spur, copper, copper, short, pin hole and copper. Model's work begins with block stem carrying of convolutional layer, which is single with 2 strides, $3 \times 3$ kernel and 32 channels output. Then, flow by batch normalization and activation ReLU. This begins with stage performance feature extraction that is low level and spatial down sampling. This changes raw RGB input in compact representation that focuses on edges and texture typical discontinuities of defects of PCB.

As, this architecture gives residual hierarchical as backbone composed of blocks (8) of residual organized in four stages. Individual block consists of 2 convolution layers along with batch normalization, $3 \times 3$ kernels and ReLU activation. Along with connection skip that assures stability of gradient and then handles effects of vanishing via DL training. When dimensions, input and output convolutional vary, then shortcut $1 \times 1$ is applied to depth channel matching. This work progressively expands feature channels from 32 to 512 meanwhile lowering spatial resolution via 2 stride convolutions ensuring multi scale presenting of both defects large structural and fine grained. This residual design gives allowance to capture local irregularities as of like copper spurs and global anomalies such as short circuits internal similar feature of hierarchy.

After this stage of final residual, network implements average adaptive pooling to sum spatial information in single 512-dimensional feature vector per image. This operation assures scale invariance and facilitates efficient global representation patterns in defect. The features pooled are then forwarded to classification head which combines dropout regularization, layer normalization, and 2 fully connected layers with activation as GELU. Then, first linear layer projects 512 - dimensional feature vector in embedding space. This is followed by dropout for overfitting avoidance. Further, linear layer maps out this embedding to 6 output neurons relating to defect classes. Then, SoftMax activation changes logits in class probabilities. Which produces final prediction vector $P \in \mathbb{R}^6$. The head design adds on label smoothing $\epsilon = 0.1$ to improve generalization and decrease overconfidence across imbalanced categories of defect.

Training is performed by utilising optimizer AdamW with annealing cosine learning rate schedule and then warm-up idea to convergence for stabilizing. Further, weighted cross entropy loss compensates class (imbalance) for Data augmentation methods like random erasing, random rotation, flipping and colour jittering are employed to enhance robustness opposing orientation illumination and variations. Trained model for epochs around 40 with precision (mixed kind) on GPU. And then achieved computationally efficient utilization of resources. Metrics of evaluation includes analysis and macro-F1 score ensuring performance, which is somewhat balanced among all classes of defect. Further, augmentation (Test-time) horizontal flipping next to it enhances consistency prediction.

For interpretability perspective, Grad-CAM visualization is implemented to last convolutional layer to highlight regions discriminative influencing every decision of classification. These heatmaps assures that network targets for relevant defect areas like copper discontinuities or broken traces. Further, projection t-SNE of learned feature embeddings showcases clear separability in categories of defect (in latent space). Validating the discriminative power of proposed model. In total, DefectNet gains strong balance among computational interpretability, efficiency and then accuracy of classification. Making it well suited for PCB automated defect inspection in environments related to industrial.

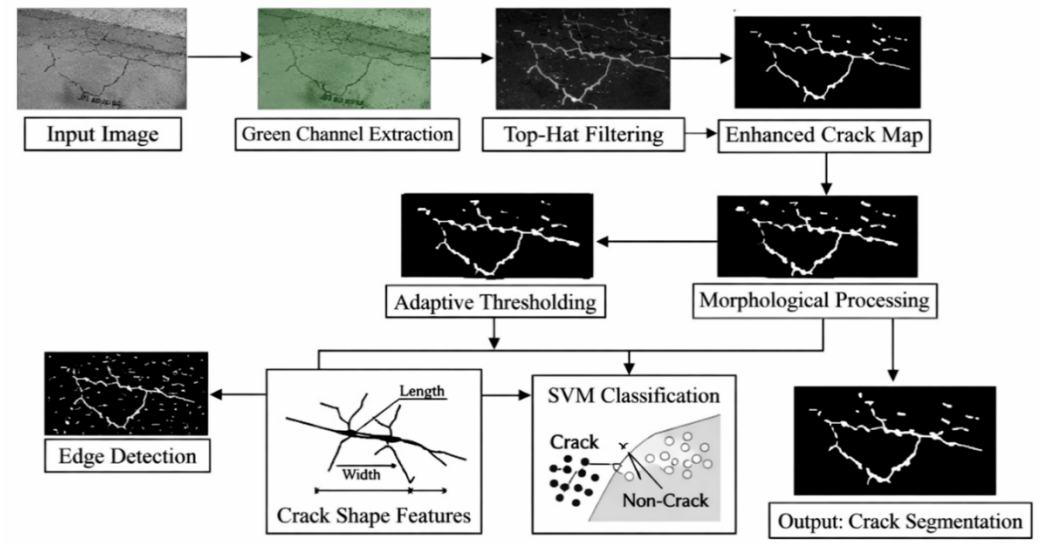


**Figure 3.** Flow diagram for Crack Forest segmentation

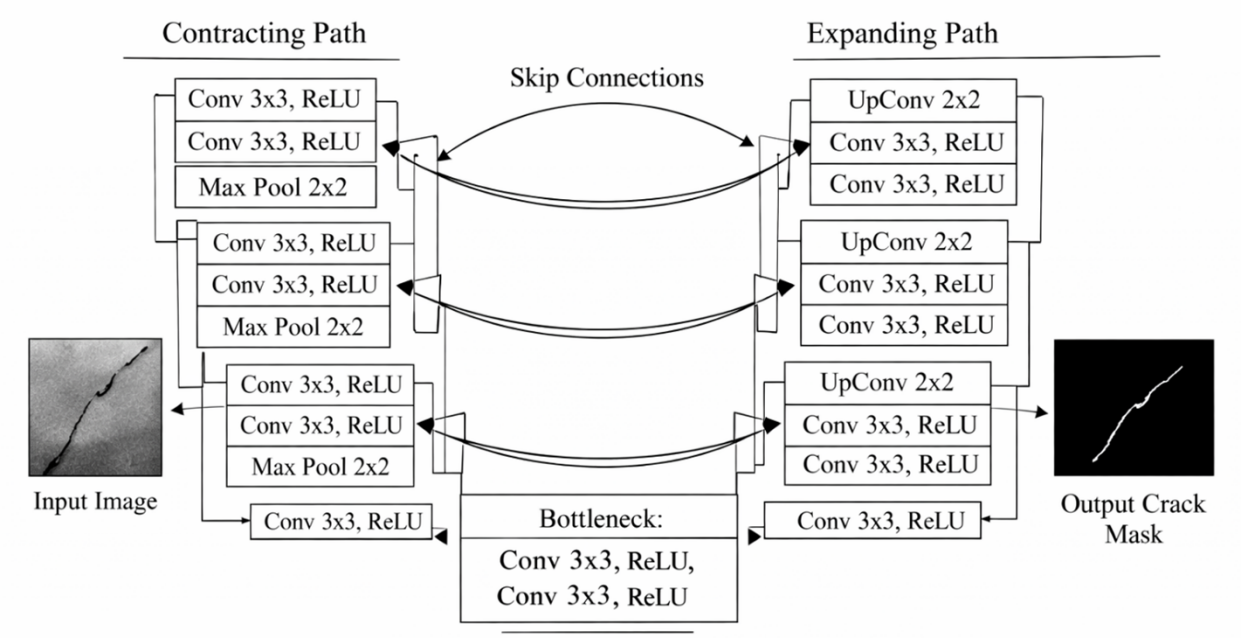


**Figure 4.** Architecture flow for proposed U-Net model

The discussed model for segmentation of crack is from-scratch U-Net architected (as shown in Figure 3 and 4) to operate on imagery pavement from dataset CrackForest. The model follows encoder–decoder classical paradigm with contracting symmetric and paths expanding, augmented by connections skip to safeguard fine-grained structural criteria of cracks. Further, contracting path contain 4 double convolutional blocks. Every individual comprising 2 successive convolutions $3 \times 3$, ReLU activation and batch normalization. These blocks progressively channel increase depth from 32 into 256 while decreasing spatial resolution via operations (max pooling). This encoding hierarchical takes both global crack continuity and local texture variations ensuring network to learn features which are discriminative among multiple scales.

At bottleneck deepest representation encodes high level semantic information via crack presence and structure. The path expanding mirrors encoder employing transposed UpConv convolutions to up sample feature maps and add up them with related encoder results via skip connections. This fusion assures that precision localization is maintained by spatial reintroducing which is lost in process of down sampling. Every individual decoder stage applies (double) convolutional blocks to refine fused features. Then, gradually decreasing channel depth back to 32. The result layer is $1 \times 1$ convolution generating single channel crack probability map that is consequently threshold to develop binary masks of segmentation.

Compound loss function combining binary cross-entropy with Dice loss and logits guided through training. Pixel-wise misclassification is penalised by logits. While dice loss focuses on overlap among predicted and crack regions ground truth effectively managing inherent class imbalance in segmentation of crack tasks. Positive class weighting factor of 10.0 is implemented to BCE to next balance sparse crack pixels against pixels dominant background. Optimization is performed utilizing AdamW with weight decay regularization and scheduled learning rate via cosine annealing with epochs warm-up to convergence stabilizing. Further, Mixed precision training along with gradient scaling assures efficiency computational on GPU.

Augmentation data plays crucial role in generalization of improvement. Random vertical and horizontal flips, rotations at multiples of 90° contrast and brightness adjustments are implemented jointly to masks and images during training. This strategy of augmentation simulates environmental diverse conditions and orientations of crack, creating model robust to variability in real world. Metrics of Evaluation include Dice coefficient, Intersection over Union (IoU), recall, pixel-level precision, and F1 score giving assessment comprehensively for segmentation quality. Next, best performing model checkpoint is chosen based on IoU validation.

For diagnostic and interpretability analysis. Further, framework gives IoU distributions, training curves, error maps, threshold sensitivity plots, and worst/ best visualizations prediction. Here, outputs highlight strengths and limitations of model's while offering insights into consistency in segmentation among diversified crack patterns. Overall, the proposed U-Net architecture reaches balance between efficiency, accuracy and interpretability. This enables well-suited for automated detection of cracks in pavement applications of maintenance.

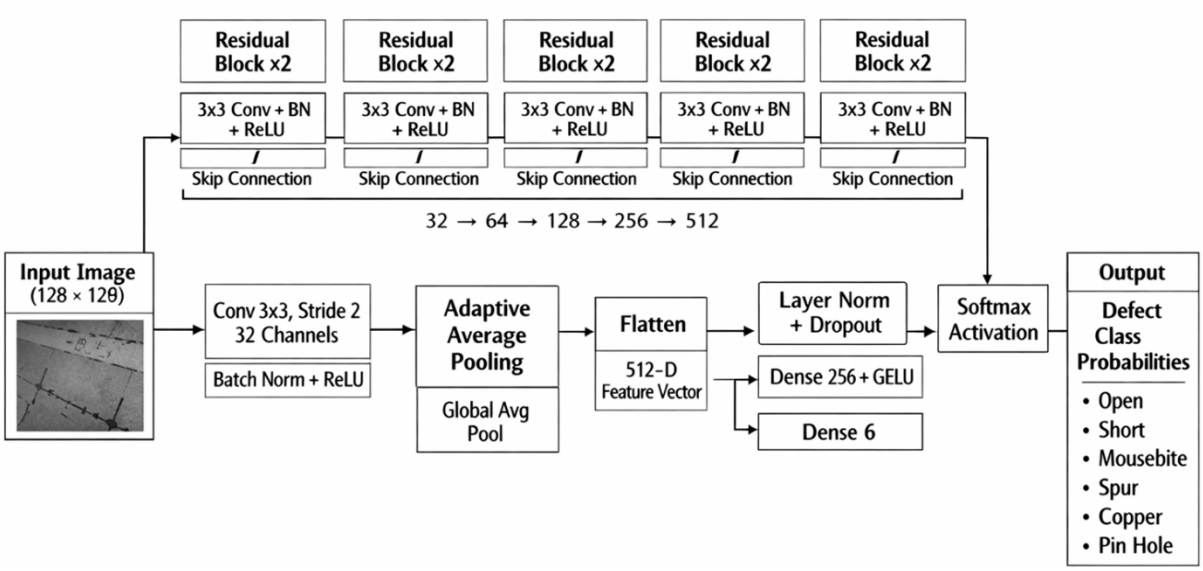


**Figure 5.** Architecture depiction of proposed model

The proposed model is custom convolutional residual neural network made for automated defect category classification in PCB - Printed Circuit Board imagery. The architecture is trained from scratch on defect cropped patches of $128 \times 128$ pixels size extracted from annotated boards in dataset DeepPCB. Further, it is architected to balance computational efficiency with discriminative high power across 6 defect types open, pin hole, mouse bite, copper, spur and short (as shown in Figure 5).

The network starts with stem block of convolutional consisting of convolution $3 \times 3$ , stride = 2, 32 channels followed by ReLU activation and batch normalization. This stage performs starts spatial down sampling and then (low-level) feature extraction. The output is passed via hierarchical backbone of residual containing 8 blocks of residual organized into 4 stages. Each block consists of two $3 \times 3$ layers of convolutional with ReLU activation and batch normalization, along with connection skip that adds on input feature map to output. When output and input dimensions differ $1 \times 1$ convolutional shortcut assures alignment of channel. The backbone expands progressively channel depth from $32 \rightarrow 64 \rightarrow 128 \rightarrow 256 \rightarrow 512$ meanwhile decreasing resolution spatial that enables multi-scale showcasing of both large structural and fine-grained defects.

After residual final stage adaptive average pooling the spatial information condensed in single 512 D feature vector per image. This ensures scale invariance. This vector is fed in head classification that implies dropout regularization at rate = 0.3, layer normalization and 2 fully connected layers with activation GELU. First linear layer transforms feature pooled in embedding intermediate. Meanwhile, second maps them to 6 output neurons related to defect categories. Activation function (Softmax) gives normalized class probabilities. Further, label smoothing $\epsilon = 0.1$ is incorporated to decrease overconfidence and then improve generalization.

Further, Training employs cross-entropy weighted loss to tackle imbalance in class with weights inversely proportional to per class counts of sample. Optimization utilises AdamW with annealing cosine and then warm-up scheduling for convergence for stability. Then, data augmentation models for random flips, color, rotations, jitter, and random erasing that enhance robustness to orientation and illumination variations. Training mixed precision with gradient scaling assures efficiency computational on GPU. Metrics for evaluation involves macro-F1 score, per-class precision-recall analysis provide balanced analysis of performance classification.

For purpose of visualization Grad-CAM focuses on regions of discriminative influencing all prediction which confirms that network focuses on areas which are defect relevant like copper discontinuities or broken traces. Further, t-SNE embedding of features learned shows differences clearly among categories of defects in latent space. Overall, DefectNet attains strong balance among accuracy computational efficiency and interpretability that makes it well suited for PCB industrial defect systems inspection.

**4.1 Experimental setup**

All approaches for this study were implemented in PyTorch and then trained from zero (no pre-trained initialization) on single GPU. Classification models utilised resolution $128 \times 128$ (DeepPCB crops, DBCC patches) or $224 \times 224$ (MTD) as input. Further, batch size of 32–64 optimizer AdamW with learning rate base as $10^{-3}$ and then weight decay as $10^{-4}$. Experimental setup also included linear-warmup added up to cosine annealing schedule then class-weighted cross entropy with smoothing of label and inference horizontal flip test time augmentation (TTA). Model's residual backbone consists of approx. parameters as 11.4 M. Datasets were partitioned by 60/20/20 stratified train/validation/test split except DBCC that uses its native partition as train/validation. The best model was selected on validation (segmentation) IoU or (classification) macro-F1 and once evaluated on held out test criteria.

**4.2 Metrics of Evaluation**

Performance classification is noted with per class recall, precision and F1-score. The weighted and macro F1 average and accuracy. For $c$ class with $TP_c$ as true positives, $FP_c$ as false positives, and $FN_c$ as false negatives:

$$\text{Precision}_c = \frac{TP_c}{TP_c+FP_c}, \quad \text{Recall}_c = \frac{TP_c}{TP_c+FN_c}, \quad \text{F1}_c = \frac{2\,\text{Precision}_c\,\text{Recall}_c}{\text{Precision}_c+\text{Recall}_c} \qquad \text{Eq. (8)}$$

F1 averages by Macro F1 among classes with weight which were equal (as shown in Eq. (8)), then creating it sensitive to performance minority class primary metric appropriate for corpora (imbalanced) studied here. Further, evaluated for segmentation with IoU (Intersection over Union), Dice coefficient and pixel-level precision/ F1/ recall.

**4.3 Magnetic-Tile-Defect (MTD)**

MTD corpus consists of 432 images among 5 classes with imbalance (as shown in Table VI). 60/20/20 split gives 87 image test set.

**Table VI.** MTD distribution of class

| Class | MT_Blowhole | MT_Break | MT_Crack | MT_Fray | MT_Free | Total |
|---|---|---|---|---|---|---|
| Images | 115 | 85 | 57 | 32 | 143 | 432 |

The from scratch framework attained best validation macro F1 score of 0.4983 and got test macro-F1 value as 0.3788 with accuracy of 0.5057 overall.

**Discussion**- On MTD performance is highly class dependent. The main defect free (MT_Free) class is recognized well (recall = 0.93, F1 = 0.74) but smallest class (MT_Fray - 6 test images) is never correctly predicted. This behavior is directly linked to the consequences of 2 compounding factors- small corpus images 432 in count total with 32 MT_Fray images give signals insufficiently for around 11.4 M parameter network trained completely from zero and then severe imbalance class leaves minority classes underrepresented irrespective of inverse frequency weighting loss. These outputs then read as lower bound baseline for pretraining free model on dataset (small). TL from large scale pretrained acts as backbone or labeled additional data would be expected to raise MTD performance substantially at framework's cost (self-contained property).

**4.4 DeepPCB (defect-type identification)**

Utilizing box guided detection to reformulation of classification (as discussed in Section 3). Annotated regions of defect were cropped in 6 class corpuses. On held out test (set of 2,003 crops) model attained test macro-F1 of 0.9945 and accuracy of 0.9945. Per-class outcomes appear in Table VII.

**Table VII.** Per-class test performance for DeepPCB

| Class | Precision | Recall | F1-score |
|---|---|---|---|
| copper | 0.9966 | 0.9932 | 0.9949 |
| mousebite | 0.9949 | 0.9949 | 0.9949 |

| | | | |
|---|---|---|---|
| open | 0.9923 | 0.9923 | 0.9923 |
| pin-hole | 0.9934 | 1.0000 | 0.9967 |
| short | 0.9900 | 0.9900 | 0.9900 |
| spur | 1.0000 | 0.9969 | 0.9985 |
| **Macro avg** | 0.9945 | 0.9946 | **0.9945** |

**Discussion-** In context to MTD, DeepPCB gives near ceiling performance among all 6 defect types. The decisive difference is data scale and balance. Further, the reformulated corpus gives 100s of well-balanced samples per class that is sufficient for (from scratch) backbone to learn features, which are discriminative (as it could be analyzed by Figure 6 ,7 and 8 depiction). Next to it, one caveat should be stated for transparency in methodological as multiple defects may initialize from physically same board, crops shared board may appear in different splits. Which may inflate score relative to strictly board partition of disjoint. Board level split is fully recommended for benchmark (leakage free criteria).

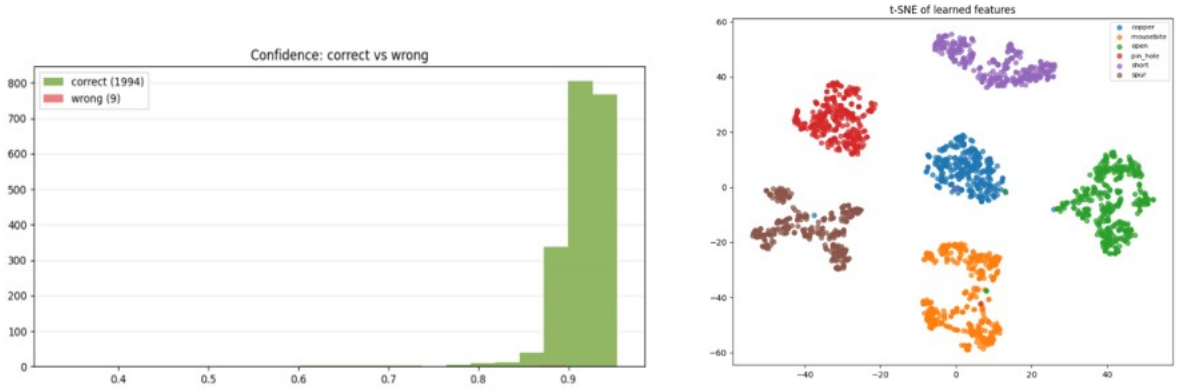


**Figure 6.** Left partition shows the confidence level for correct vs wrong, while right side picture shows the t-SNE of learned features

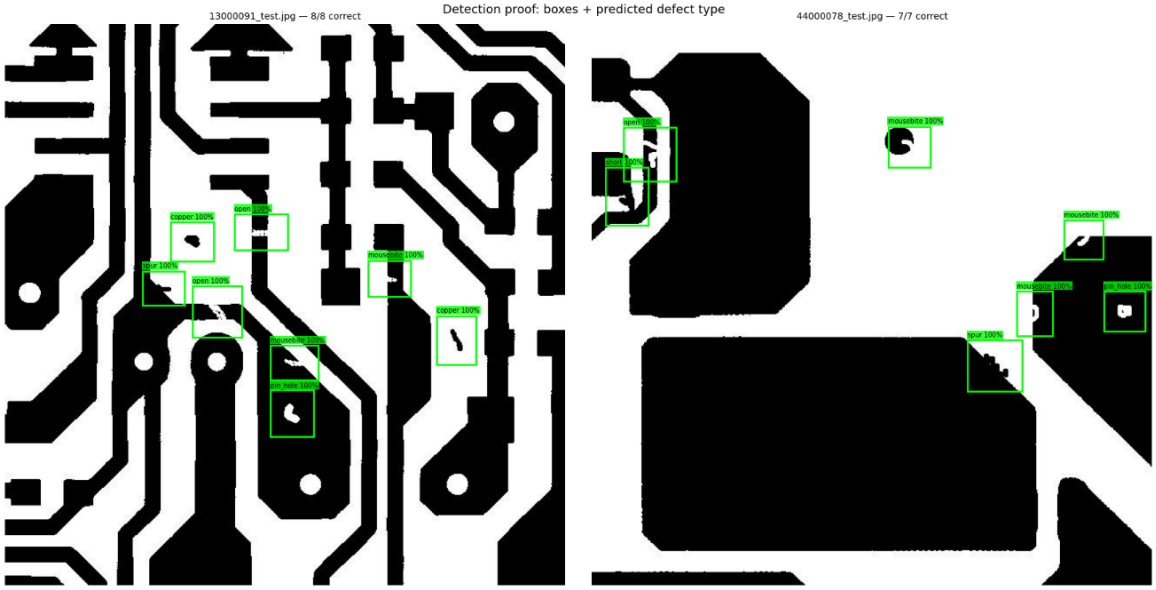


**Figure 7.** Detection proof- which shows defected area and the type of defect

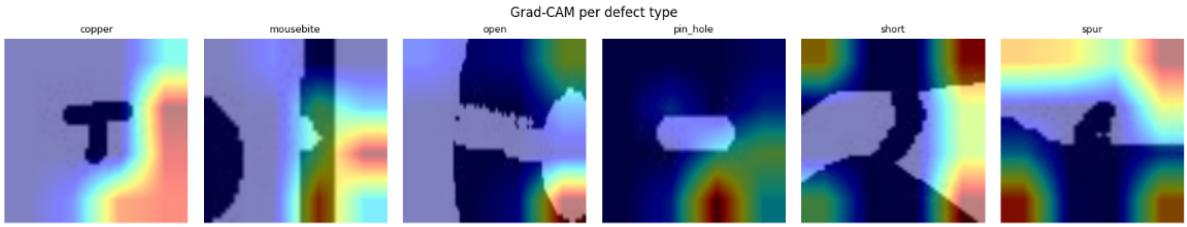


**Figure 8**. For various types, Grad Cam highlighting the type of defect

### 4.5 DBCC Bridge Crack for binary

DBCC corpus consists of training sample as 50,000 and validation with 7:1 imbalance approx. favouring sample *no-crack*, validation native set was bifurcated in validation disjoint and subsets for test. Validation earlier shown rapid learning with validation macro-F1 reached 0.982 (in first epoch) and final test run generated near perfect performance-with macro-F1 as 0.993, approx. 0.992 / 0.994, ROC-AUC = 0.999 for per-class F1 (crack / no-crack) and precision average (crack) as 0.998 (as presented in Figure 9 and 10). Confidence distributions focus at high scores and visualizations of feature space indicating clear separation among no-crack and crack examples. This confirms that model learned discriminative features highly for this task (binary) at data scale which is available. These outcomes depict that given labelled sufficient examples even with imbalance class compact from base classifier can attain nearby performance ceiling on crack detection (binary).

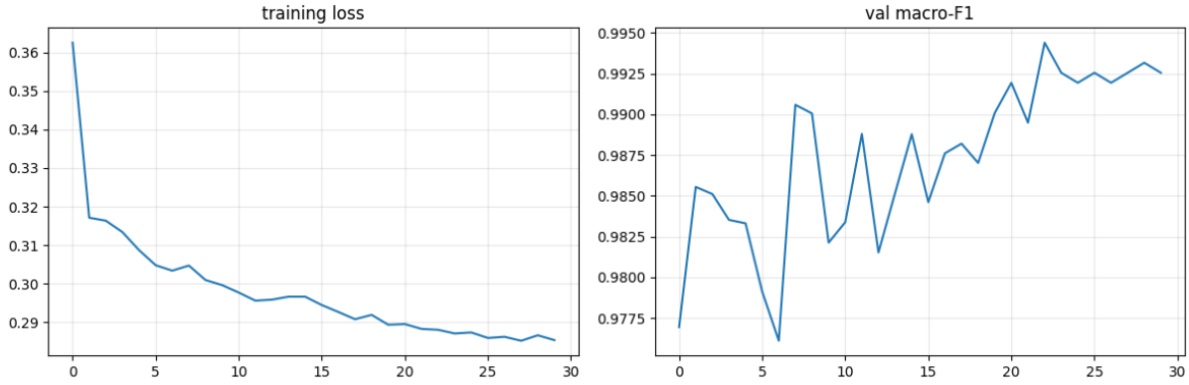


**Figure 9.** Left graphs give the training loss values while right gives the Val macro F1 values

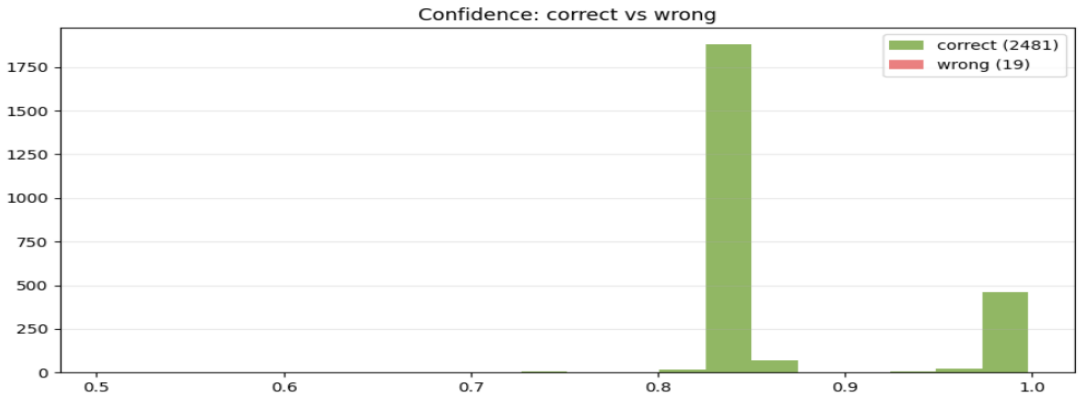


**Figure 10.** Depiction of confidence value for correct vs wrong results

### 4.6 CrackForest (in relation to segmentation)

Segmentation of pixel-level crack was implemented with from-scratch trained U-Net using BCE plus Dice objective. The set of validation guided threshold operating of 0.30 and final test metrics counted mean IoU of approx. 0.50–0.55 and Dice ≈ 0.67 at pixel-F1 of approx. 0.68. Training loss minimized validation IoU plateaued in range low-0.5. Overlays of qualitative and maps of error shows various predictions which are strong best IoU of approx. 0.64–0.66 alongside cases of failure worst IoU approx. 0.01–0.36 (as shown in Figure 11, 12, 13, 14 and 15). This produced balance regions of true positive with limited false negatives false positives. These results of segmentation are constant with expectations for training on Crack Forest and focuses on greater limitation of pixel-level tasks under small sample size data in comparison with well-balanced large, classification corpora.

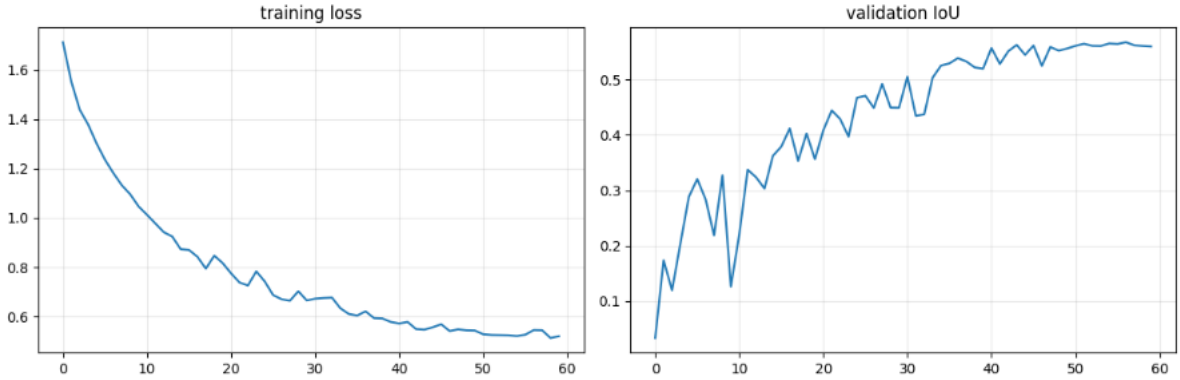


**Figure 11.** Output graphs for training loss (on left side) and validation IoU (on right side)

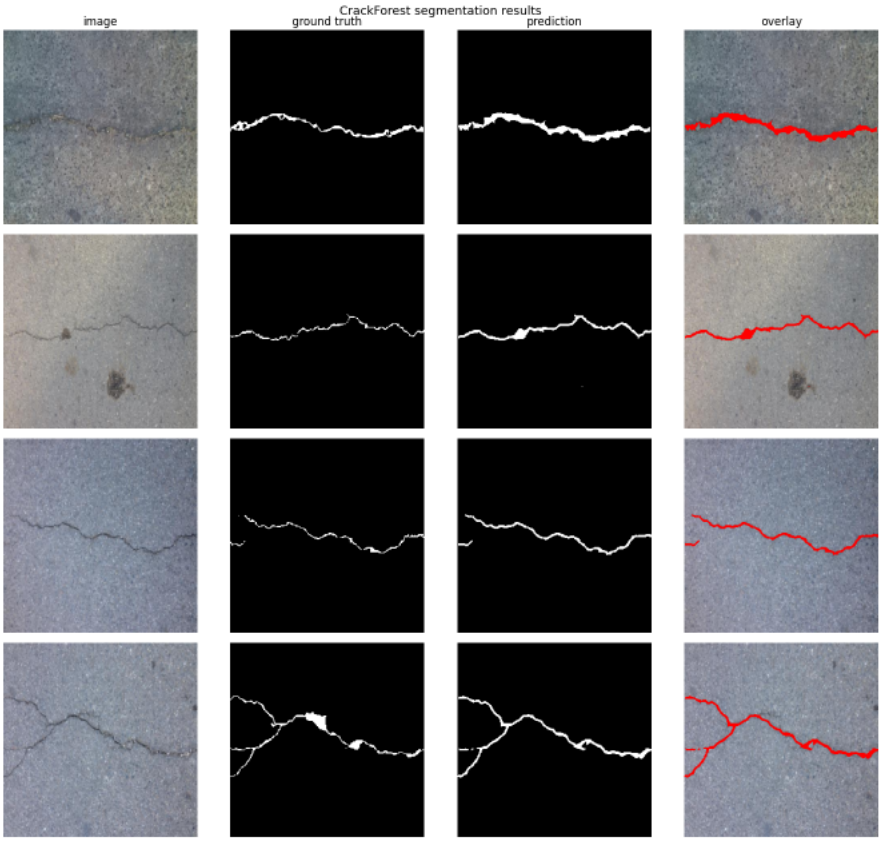


**Figure 12.** Results of segmentation of crackforest (crossing various stages)

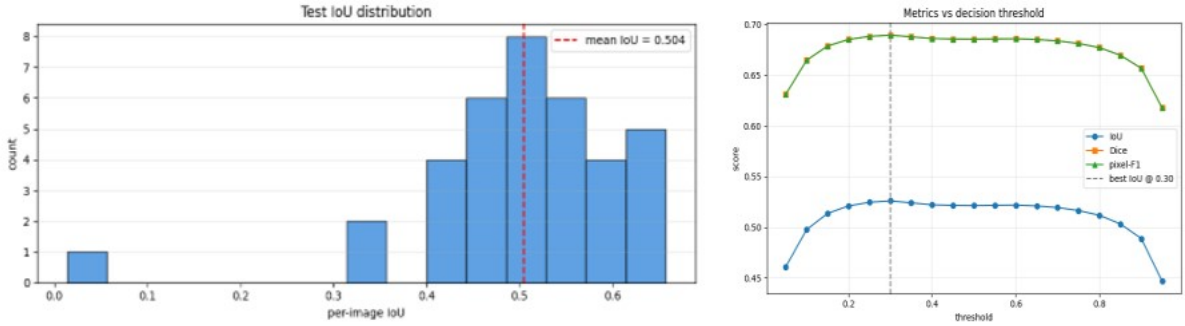


**Figure 13.** Graph (on left) shows Test values of IoU distribution and right graph shows Metrics vs decision threshold

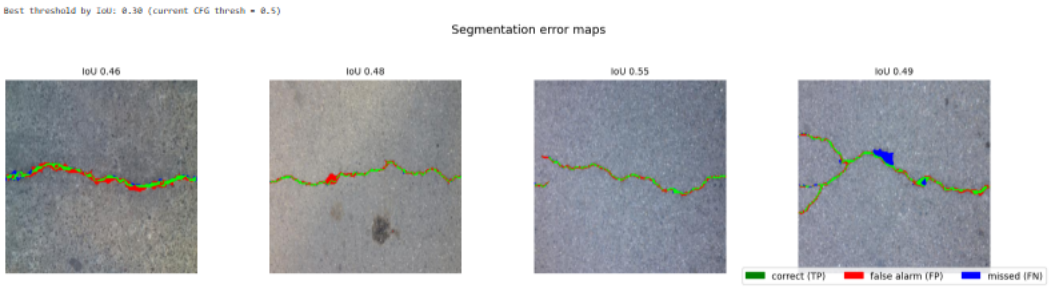


**Figure 14.** Segmentation error maps and various IoU values

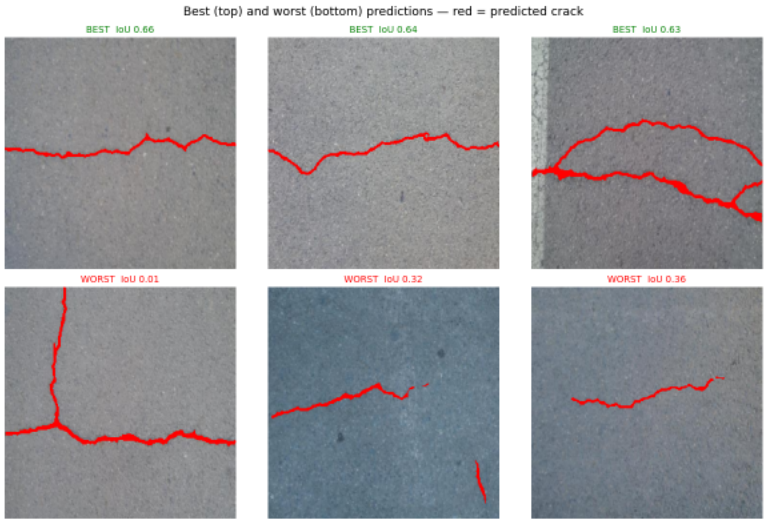


**Figure 15.** Best values depicted on top and worst prediction(below) with red line for prediction of cracks

### 4.7 Cross-dataset discussion

The outcomes reveal pattern consistency - similar from-scratch model attains performance (near perfect) on well-balanced, large corpora (DBCC, DeepPCB) but only perform modest on datasets which are smaller or class imbalanced (CrackForest, Magnetic-Tile-Defect) (as shown in Table VIII). This contrast is empirical central finding. The model's effectiveness is handled primarily by class balance and data scale rather than by capacity architectural. Where sufficient balanced data exists, a compact, pre-training-free model is effective (highly). Here, data are heavily or scarce skewed, lack of pre-trained priors turns into the limiting factor. These observations give way to 2 practical directions for enhancing robustness in regime low data- combining optional backbone pretrained to give stronger priors in situation when labeled data are low in number and pursue specific targeted collection of data and then augmentation of class balanced to minus minority class scarcity and overall improving generalization.

**Table VIII.** Output summary of test performance.

| Dataset | Task | Primary metric | Value |
|---|---|---|---|
| Magnetic-Tile-Defect | 5-class classification | macro-F1 | 0.3788 |
| DeepPCB | 6-class classification (crops) | macro-F1 | 0.9945 |
| DBCC | binary classification | macro-F1 | *0.993* |
| CrackForest | segmentation | IoU / Dice | *0.504 / 0.67* |

Unlike approaches based on conventional CNN/U-Net which perform computationally intensive on feature extraction and dense level of pixel prediction, this proposed models adopts lightweight crop level classification approach which focuses on learning discriminative showcasing from regions related to defect(centered). Novelty of proposed methodology lies in this classification and decoupling of localization. Which reduces computational and memory requirements related to dense prediction. Meanwhile, providing efficient solution for recognition of defect. DeepPCB annotations are especially utilized to target defect containing areas and generate representative crops for classification. As these annotations gives reliable supervision spatially for constructing defect focused, training samples. Because proposed model performs classification on (annotation guided) crops rather than independently predicting of pixel level masks. Its localization capability is limited inherently by giving crop boundaries. Hence, not same as U-Net based segmentation methods that can generate fine grained defect contours, this proposed architecture prioritizes computational efficiency and also accurate crop level defect recognition. Thus, making it more suitable for resource constrained deployment in practical manner.

## 5. Conclusion

To conclude this study, the proposed development evaluated single, self-contained DL model for analysis of automated surface defect detection. Which is motivated by safety, critical requirement to inspect infrastructure and industrial components are ageing more reliably than subjectively, slow and allows manual hazardous inspection. This work compact backbone residually with regularized of classification head and then symmetric U-Net model was trained from start and applied under protocol (commonly used) for 4 heterogeneous types covering classification for multi class, crack (binary) detection and segmentation at level of pixel utilizing imbalance aware objectives and data (format agnostic) ingestion. Intellectually, Model attains good performance on balanced and large corpora (DBCC, macro-F1 $\approx$ 0.993, DeepPCB, macro-F1 = 0.9945) meanwhile giving more results on strongly small or skewed datasets (CrackForest, IoU approx. 0.50–0.55, Dice approx. 0.67; Magnetic-Tile-Defect, macro-F1 = 0.3788;). The findings are practical and in absence of data scale, pre-training and balance of class dominate gives more than capacity which are architectural. Instead of models advocating singly for each setting. The contribution depicts that lightweight, (pretraining free) pipeline can tackle inspection multiply clarifying paradigms, specifically when there are additional requirements of counts. Future work could be done in direction for combining optional encoders (pre trained) for adopt board-disjoint, low-data regimes and evaluations of multi seed for statistical rigor, emphasizing augmentation of class balanced and further target collection of data for rare types of defects, exploring self-supervised and semi curricula to leverage data, which is unlabeled and pipeline extended in direction for end to end detection and deployment.

## References

[1] Y. Chen, Y. Ding, F. Zhao, E. Zhang, Z. Wu, and L. Shao, "Surface defect detection methods for industrial products: A review," *Applied Sciences*, vol. 11, no. 16, Art. no. 7657, 2021, doi: 10.3390/app11167657.

[2] D. Martin, S. Heinzel, J. Kunze von Bischhoffshausen, and N. Kühl, "Deep learning strategies for industrial surface defect detection systems," *arXiv preprint* arXiv:2109.11304, Sep. 2021.

[3] Q. Jin and L. Chen, "A survey of surface defect detection of industrial products based on a small number of labeled data," *arXiv preprint* arXiv:2203.05733, Mar. 2022.

[4] Y. Ma, J. Yin, F. Huang, Q. Li, *et al*., "Surface defect inspection of industrial products with object detection deep networks: A systematic review," *Artificial Intelligence Review*, vol. 57, Art. no. 333, 2024, doi: 10.1007/s10462-024-10956-3.

[5] Y. Cheng, Y. Cao, H. Yao, W. Luo, C. Jiang, H. Zhang, and W. Shen, "A comprehensive survey for real-world industrial defect detection: Challenges, approaches, and prospects," *arXiv preprint*, 2025.

[6] X. Wang, Y. Liu, H. Zhang, *et al*., "Review of surface-defect detection methods for industrial products based on machine vision," *IEEE Access*, vol. 13, 2025, doi: 10.1109/ACCESS.2025.3571297.

[7] G. Qin, C. Liu, H. Li, and K. Xu, "A review of deep learning methods for metal surface defect detection," *Steel Research International*, 2026, doi: 10.1002/srin.202501060.

[8] L. Zhao, Y. Yuan, and Y. Wu, "A survey of surface defect detection in machine vision: Addressing core challenges, methodologies, and dataset analysis," *Computers, Materials & Continua*, vol. 88, no. 2, 2026, doi: 10.32604/cmc.2026.080232.

[9] Y. Cheng, Y. Cao, H. Yao, *et al*., "A comprehensive survey for real-world industrial surface defect detection: Challenges, approaches, and prospects," *Journal of Manufacturing Systems*, vol. 84, pp. 152–172, 2026.

[10] F. Shang, H. Chen, B. Sun, *et al*., "A review of deep learning-based surface defect detection techniques for strip steel," *Measurement*, 2026, doi: 10.1016/j.measurement.2026.121905.

[11] Q. Qin, A. S. M. Khairuddin, N. Idros, *et al*., "Hierarchical depth-aware YOLO for efficient metal surface defect detection," *Scientific Reports*, vol. 16, 2026.

[12] E. E. Somtochukwu and N. Rijal, "Real-time industrial defect detection on edge hardware using fine-tuned YOLOv8," *arXiv preprint*, 2026.

[13] P. Bergmann, M. Fauser, D. Sattlegger, and C. Steger, "MVTec AD—A comprehensive real-world dataset for unsupervised anomaly detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2019, pp. 9584–9592.

[14] K. Roth, L. Pemula, J. Zepeda, B. Schölkopf, T. Brox, and P. Gehler, "Towards total recall in industrial anomaly detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2022, pp. 14318–14328.

[15] M. Rudolph, T. Wehrbein, B. Rosenhahn, and B. Wandt, "Asymmetric student-teacher networks for industrial anomaly detection," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, 2023, pp. 2592–2602.

[16] P. Mishra, R. Verk, D. Fornasier, F. Piciarelli, and G. Foresti, "DRAEM: A discriminatively trained reconstruction embedding for surface anomaly detection," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 8330–8339.

[17] T. Defard, A. Setkov, A. Loesch, and R. Audigier, "PaDiM: A patch distribution modeling framework for anomaly detection and localization," in *Proc. Int. Conf. Pattern Recognit. (ICPR)*, 2021, pp. 475–489.

[18] M. Salehi, N. Sadjadi, S. Baselizadeh, M. Rohban, and H. Rabiee, "Multiresolution knowledge distillation for anomaly detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2021, pp. 14902–14912.

[19] V. Zavrtanik, M. Kristan, and D. Skočaj, "DSR: A dual subspace re-projection network for surface anomaly detection," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2022, pp. 539–554.

[20] X. Tao, X. Gong, X. Zhang, S. Yan, and C. Adak, "Deep Learning for Unsupervised Anomaly Localization in Industrial Images: A Survey," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, Art. no. 5018021, 2022, doi: 10.1109/TIM.2022.3196436