STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **x**, Month 2025, pp 0–19.
Published online in International Academic Press (www.IAPress.org)

Defect Inspection on Metal Production Using Image Processing

Omnia Khalaf Kamel, Mohamed Hassan, MARY MONIR SAEID*

Faculty of computers and Artificial intelligence, Fayoum university. Egypt

Abstract Enhanced product performance and quality control are essential for advanced industrial systems. Nonetheless, various surface defects adversely affect the product's performance, as well as its aesthetics and functionality. This study investigates the detection and classification of surface defects in industrial materials using advanced deep learning techniques. This paper focuses on quantifiable targets based on baseline performance in manufacturing environments through the utilization of deep learning models applied to the NEU Surface Defect Database. Convolutional neural network (CNN)-based approaches as MobileNetV2 Model and ResNet50V2 Model are employed, leveraging image processing techniques to identify six common types of surface defects. The MobileNetV2 Model is implemented and tested to obtain high accuracy besides specifying the resulting limitations. On the other hand, the ResNet50V2 Model is enhanced by adding more layers to overcome overfitting and the resulted limitations of the MobileNetV2 Model, achieving accuracy of 95% for classifying each defect. These results demonstrate that the proposed framework achieves high accuracy in defect detection, proving that deep learning techniques can significantly improve quality control in manufacturing processes, reducing manual inspection efforts and minimizing errors.

Keywords Steel Surface Defects, convolution neural networks (CNNs), MobileNetV2 Model, ResNet50V2 Model

DOI: 10.19139/soic-2310-5070-3181

1. Introduction

The drive for advanced industrial systems necessitates improved product performance and heightened production quality control. Metal surface flaws present a particularly significant challenge, negatively impacting not only appearance and usability but also critical performance metrics. The detection of these defects is difficult due to their often-small size and the wide variety of types that can occur. Key defects that compromise material integrity include vacancies, rolled-in scale, patches, crazing, pitted surfaces, and inclusions. A vacancy is a point defect in a crystal structure where an atom is absent from its typical location, causing a local increase in strain energy due to the disruption of the normal atomic arrangement and slight positional misalignment [1]. Rolled-in scale flaws manifest as rough, black spots or streaks that detrimentally impact mechanical properties, corrosion resistance, and surface quality, frequently requiring additional finishing like pickling or grinding for removal [2]. Patches refer to areas exhibiting irregular texture, color, or finish, often appearing as spots or discoloration caused by inconsistent heating/cooling, improper surface treatments, contamination, or oxidation; while sometimes cosmetic, they can indicate serious underlying issues like corrosion or poor coating adherence [3]. Crazing is a network of fine cracks on the surface, and a pitted surface is characterized by small cavities or holes, both of which severely weaken the surface and can act as initiation sites for catastrophic cracks. Finally, inclusions are internal flaws involving non-metallic particles or impurities like oxides or sulfides inadvertently introduced during melting, casting, or forming; these alter structural homogeneity and impair mechanical qualities and performance [4]. Automated flaw identification based on computer vision has been fueled by machine learning (ML). From early concepts

*Correspondence to: MARY MONIR SAEID (Email: mmh04@fayoum.edu.eg). Department of computer science, Faculty of computers and Artificial intelligence, Fayoum university. Egypt

ISSN 2310-5070 (online) ISSN 2311-004X (print)

to advanced deep learning (DL) and neural networks, ML has evolved substantially since the mid-20th century [5]. While classifiers like Support Vector Machines (SVM) function by distinguishing defective from non-defective regions, deep learning techniques are now the most promising, capable of identifying flaws imperceptible to conventional algorithms. By learning patterns from training data, a neural network can categorize new inputs. Recent years have seen many proposed deep learning quality assessment systems based on segmentation [6]. Here, Convolutional Neural Networks (CNNs) are frequently used for automatic hierarchical feature extraction, as their parameter-efficient architecture with sparse connectivity effectively reduces model complexity. This capability stems from the convolutional layer, the main computational component of a CNN, responsible for extracting features from input data such as color images. The input image has three dimensions (height, width, and depth for RGB). A small 2D filter (kernel), typically 3×3, slides across the image and performs convolution by computing dot products between the filter and local regions of the image. This process generates a feature map that highlights detected features. As the filter moves across the image using a defined stride, it shares the same weights at all locations (parameter sharing). The output size is controlled by three hyperparameters: the number of filters (determines output depth), stride (controls how far the filter moves), and padding (valid, same, or full), which affects output dimensions. During training, filter weights are updated via backpropagation. After convolution, a ReLU activation function is applied to introduce nonlinearity into the network. Digital image processing is a key computational technology in this domain, used in advanced inspection techniques—for example, to accurately calculate rust percentages on steel coatings [7]. Computer vision and artificial neural networks are integrated to automate the visual inspection of surface flaws on rolled steel, a crucial need as increasing industry throughput makes manual inspection a bottleneck. These systems assess images by classifying surface texture or segmenting features, combining texture analysis with mathematical models to meaningfully characterize surfaces and differentiate between defective and sound areas [8].

1.1. Challenges in Steel Surface Defect Detection

Detecting and diagnosing defects in metal and steel surfaces using machine learning, particularly convolutional neural networks (CNNs), is a critical area of research in industrial quality control. Several studies from reputable scientific journals have addressed the challenges associated with this task. Below is a summary of key findings:

- Human inspection faces limitations due to its inefficiency, inconsistency, and human error, particularly in large-scale industrial applications [9], which can reduce accuracy and reliability in defect detection, potentially leading to quality control failures.
- CNN models, particularly those requiring high-performance GPUs, are computationally intensive, leading to increased operational costs and limiting accessibility for smaller companies in real-time manufacturing environments [10].
- CNNs face challenges in extracting meaningful features from complex metal surfaces, particularly in the presence of noise or minor variations, which can lead to false positives or missed defects, reducing the reliability of automated quality control systems [11].
- Steel surface defect datasets often face limitations due to class imbalance, leading to biased learning models that perform well on common defects but fail to accurately detect rare defects, reducing overall effectiveness [12].
- Generalization issues in manufacturing environments can lead to extensive re-training and adaptation, increasing development time and complexity [13], as models trained on one dataset may not effectively generalize to different environments.
- Defect detection models in real-time production settings face challenges due to latency issues and slow inference speeds, which can affect production efficiency by causing defective products to be processed further down the supply chain [14].
- Energy-efficient AI models are crucial for long-term industrial applications due to their high power consumption and environmental impact, which can increase the operational costs and environmental footprint of AI-driven solutions [15].

1.2. Objectives and Contribution

- Strategic Fine-Tuning Protocol: Targets the last 50 layers of ResNet50V2 to enhance feature adaptation while preserving low-level features from ImageNet, optimizing transfer learning for small datasets.
- Custom Classification Head: Replaces the original model head with a bespoke sequential block to minimize overfitting and effectively learn optimal feature combinations for a 6-class classification task.
- Performance Benchmarking Framework: Develops a framework for comparing MobileNetV2 and ResNet50V2 models under consistent data and training conditions, focusing on the accuracy-efficiency balance essential for industrial applications.
- Reproducibility Pipeline: Implements a detailed pipeline for data handling, augmentation, training, evaluation, and prediction for both models, serving as a valuable reference for future research and engineering endeavors.

Therefore, the primary contribution is the provision of a reproducible, carefully tuned benchmark and a clear analysis of the practical considerations (like the balance between fine-tuning depth and overfitting risk) for applying deep learning to industrial visual inspection. This offers significant value to researchers and practitioners aiming to deploy reliable systems in real-world settings."

2. Related Work

The detection and classification of surface defects in steel have evolved significantly over the past decades. This section reviews key methodologies in the field, categorizes them by their technical approach, highlights their limitations, and identifies persistent research gaps that motivate the present study.

2.1. Traditional Image Processing and Classical Machine Learning Methods

Early approaches relied heavily on handcrafted features and classical machine learning algorithms. These can be broadly categorized into statistical, spectral, model-based, and early learning techniques, each with distinct operational frameworks and constraints.

- **Statistical & Spectral Methods**: Foundational techniques analyzed pixel intensity distributions or transformed images into frequency domains. As surveyed by Sun et al. [16], these include thresholding, edge detection (e.g., Sobel, Canny), Local Binary Patterns (LBP), Gray-Level Co-occurrence Matrices (GLCM), and transforms like Fourier, Gabor, and Wavelet. For instance, Luo et al. [17] categorized defects by analyzing image histograms, a method highly dependent on contrast. While computationally efficient, these methods are sensitive to noise, non-uniform illumination, and scale changes. They often fail when defect-background intensity differences are minimal, and advanced variants like Multiscale Geometric Analysis struggle with parameter tuning and feature redundancy [[17],[16]].
- **Model-Based Methods**: These approaches project texture into low-dimensional models. Techniques include Markov Random Fields (MRF) for spatial correlation and Active Contour Models (ACM) for boundary precision [[17],[16]]. However, MRF models are unsuitable for global texture analysis and small defects, while ACMs face convergence issues and high computational costs, making them less viable for real-time application [17].
- **Classical Machine Learning**: Supervised algorithms like Support Vector Machines (SVM) were commonly paired with handcrafted features. Lu and Chen [18] developed a multi-stage framework using a Local Multi-Neural Network (Lc-MNN) with wavelet features and a custom SCV selection algorithm, achieving high precision on bearing defects. Similarly, a multi-class SVM system in [19] achieved 87–94% accuracy using geometric and grayscale features. However, these methods are constrained by their dependence on manually engineered features, which may not generalize across defect types. They also face challenges with complex real-world conditions, such as the micro-defect detection and initial segmentation dependency noted in [18]. Other methods, such as the thresholding and filtering applied to cylindrical surfaces in [20], achieved low

error rates but with significant over-detection (overkilling), indicating a critical trade-off between sensitivity and specificity.

These traditional methods are collectively limited by their dependence on handcrafted features, sensitivity to imaging conditions (e.g., lighting, contrast), limited generalization to 3D or complex surfaces, a tendency toward over-detection or missed defects, and often prohibitive computational demands for real-time, online inspection [[17], [13], [16]].

2.2. Early Deep Learning Approaches

The advent of deep learning, particularly Convolutional Neural Networks (CNNs), marked a shift toward automated feature learning, offering improvements in accuracy but introducing new challenges.

- **CNN Architectures for Classification**: Studies demonstrated the power of CNNs to learn features directly from data. Lee et al. [21] proposed a VGG-like 17-layer CNN integrated with Class Activation Maps (CAM) for interpretability, achieving 99.44% accuracy on the NEU dataset. This represented a significant leap over traditional methods like SVM with GLCM/HOG features. However, the model was designed for single-defect classification per image and remained sensitive to environmental noise and dataset-specific hyperparameters [21]. Other works, such as the classical CNN applied to rail surfaces in [22], achieved low error rates but highlighted overfitting risks on small datasets.
- **Hybrid and Specialized Frameworks**: Researchers combined deep learning with hardware or algorithmic innovations to address specific industrial challenges. Chen et al. [20] developed the Baru-Net for reflective surfaces, integrating symmetrical lighting, a U-Net with CBAM and ASPP modules, and artificial data augmentation to achieve 98.3% accuracy. While effective, this framework was limited to a small set of defect types, relied on a very small original dataset and manual defect synthesis, and its hardware automation was incomplete [20]. A hybrid approach in [23] combined multi-illumination imaging with a shallow network, showing variable performance across defect classes.

Limitations Summary: Early CNNs showed promise but often suffered from overfitting on small datasets, limited accuracy on specific defect classes, high computational demands, and insufficient robustness to real-world variations in illumination and surface texture [[21], [20]]. They also struggled with multi-defect scenarios and required significant effort in data preparation and model tuning.

2.3. Recent Advances: CNN Enhancements and Transformer-Based Models

Recent research has focused on enhancing CNN architectures for efficiency and accuracy and exploring the potential of transformer-based models.

- **Enhanced CNN Architectures**: Innovations often involve attention mechanisms and multi-scale feature fusion. Wang et al. [19] enhanced YOLO-v5 with a Multi-Scale Explore Block and a Spatial Attention mechanism, achieving a 72% mAP on NEU-DET at high speed (192.3 FPS). While meeting real-time requirements, the model showed inconsistent performance across defect types (e.g., struggling with crazing) and incurred increased computational cost [19]. Other studies, like Defect-Net [24] and the use of DenseNet201 via transfer learning [25], achieved high accuracy (up to 99.51%) but were noted as computationally expensive and time-consuming to train.
- **Transformer-Based Models**: Models like HCT-Det [26] introduced hierarchical self-attention mechanisms for better feature alignment, achieving competitive performance. However, these models typically come with high computational complexity, reduced feature resolution for small defects, and increased training and deployment challenges, making them less suited for resource-constrained environments [26].
- **Robustness and Corner Case Detection**: Addressing model reliability, Ouyang et al. [27] developed modified Distance-Based Surprise Adequacy (DSA) metrics to detect misclassified and adversarial samples by analyzing neuron activation traces. Although effective, these tools are limited to classification tasks, incur quadratic computational complexity, and their efficacy is dependent on the specific model layer chosen for analysis [27].

While recent models achieve higher accuracy, they frequently entail greater computational complexity, are prone to overfitting without extensive data augmentation, and struggle with small-scale defects. There remains a significant trade-off between accuracy, speed, and parameter efficiency, with many advanced models not being optimized for real-time, edge-device deployment [[19], [26]]. Furthermore, ensuring robustness against corner cases and adversarial noise adds another layer of computational overhead [27].

2.4. Research Gaps and Motivations

Despite these advancements, several critical gaps remain, as highlighted by the limitations of the reviewed work:

- **Efficiency vs. Accuracy Trade-off**: Many high-accuracy models (e.g., deep CNNs, transformers) are computationally heavy, limiting their use in real-time or edge-device applications. Lightweight models like MobileNetV2 have been underexplored for steel defect detection, especially in direct comparison with deeper networks. Even enhanced real-time models like [19] see performance gaps on specific defect types when computational constraints are applied.
- **Generalization and Real-World Validation:** Most studies are evaluated on controlled benchmark datasets like NEU-DET. There is a lack of systematic validation on real-world, out-of-distribution images with varying lighting, angles, or material types. The dependence on tailored hardware setups [20] or specific feature sets [18] further questions generalizability.
- **Comprehensive Performance Analysis**: Few works provide a holistic comparison of model families (e.g., lightweight vs. deep CNNs) that includes not only accuracy but also training time, inference speed, parameter efficiency, and robustness metrics like those proposed in [27].
- **Effective Strategies for Limited Data**: Although techniques like artificial data augmentation [20] and transfer learning are used, optimal fine-tuning strategies for industrial datasets with limited samples and class imbalances are not thoroughly investigated or standardized. The field also lacks well-recognized standard datasets and protocols, hindering fair comparison [16].

This work directly addresses the unresolved efficiency-accuracy trade-off for industrial deployment by providing a direct, comprehensive comparison between a lightweight architecture (MobileNetV2) and a deeper, high-accuracy architecture (ResNet50V2).

3. Proposed Framework

The proposed framework outlines a step-by-step process for building a Steel Surface Defects Detection model. First, Dataset Creation involves gathering and labeling data process. Next, Data Preprocessing cleans and prepares the data. Feature extraction transforms raw data into significant patterns utilizing methods such as pre-trained algorithms. Custom classification layers are subsequently incorporated to customize the model for the specific tasks. Finally, Model Training and Fine-Tuning optimizes the model's performance, followed by Evaluation and Prediction where 2 different models are tested and the most accurate one is deployed for real-world use. as illustrated in Figure 1.

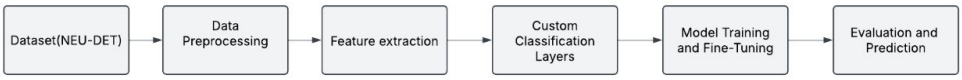


Figure 1. Proposed Steel Surface Defects framework

3.1. Northeastern University (NEU) Dataset

The Northeastern University (NEU) surface defect database is extensively utilized in computer vision and machine learning for industrial defect detection [38], especially in metal surface inspection. It comprises photographs

of steel surfaces exhibiting six prevalent defect types: rolled-in scale (RS), patches (Pa), crazing (Cr), pitted surface (PS), inclusion (In), and scratches (Sc). Figure 2 presents sample photographs of these defects, illustrating considerable discrepancies in appearance among intra-class errors. Inter-class defects exhibit features such as pitted surfaces, crazing, and rolled-in scale. The grayscale of intra-class faults fluctuates due to material and illumination influences. The database encounters two challenges: considerable discrepancies in appearance among intra-class faults and analogous characteristics among inter-class defects, along with alterations in material and illumination effects on defect images. The dataset, structured with image labels, is used for classification and segmentation tasks, and can be used for supervised learning. Most popular applications are quality control in industrial manufacturing, defect classification using deep learning models, and anomaly detection. To guarantee high-quality and representative photos of surface flaws, the dataset creation procedure consists of several steps. These specific steps involved in creating the dataset, as well as its characteristics, potential biases, and mitigation techniques, are discussed below:

- **Dataset Features**:According to the previous steps involved in creation of dataset, The main features of the NEU Surface Defect Dataset are concluded in table 1.

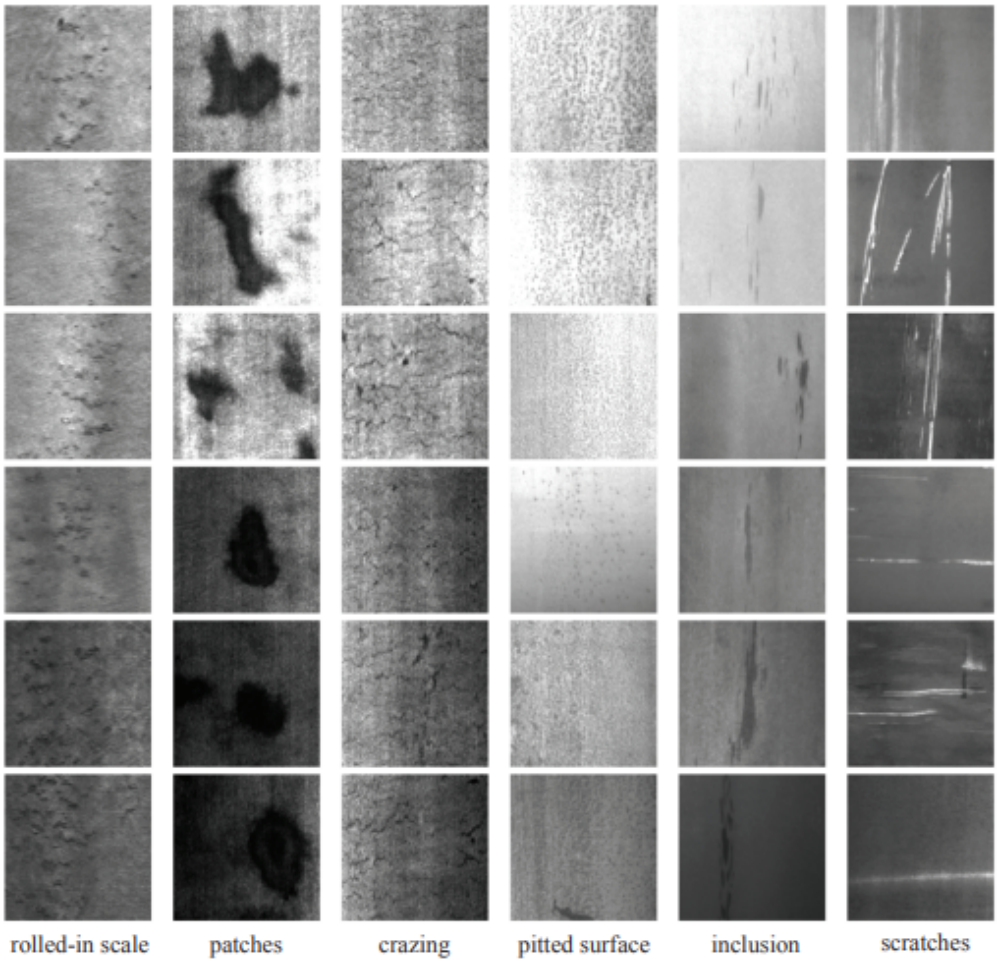


Figure 2. NEU surface defect database samples of six common surface defect types. One sample image from each of the 300 class samples is displayed in each row.

Table 1. NEU Surface Defect Dataset features

Feature	Description
Image Resolution	200 x 200 pixels (standard size)
Color Format	Grayscale or RGB
Number of Classes	6 (Crazing, Inclusion, Patches, Pitted Surface, Rolled-in Scale, Scratches)
Total Images	1,800 images (300 per defect type)
Annotation	Each image is labeled with its defect type
Application	Machine learning models for defect classification and segmentation

The NEU Surface Defect Dataset's Bias: Not withstanding its usefulness, the dataset could include biases that affect how well the model performs:

- **An imbalance in data:** To avoid imbalance at the dataset level, the dataset contains 300 photos for each defect type, or an equal number of images for each class. However, some flaws are more common than others in real-world situations, which is not reflected in the dataset.
- **Insufficient Diversity:** Because the photos were shot in controlled settings, they might not accurately depict the variances found in actual industrial settings (e.g., varied lighting conditions, camera angles). The model might have trouble with photos that are very different from the dataset if it is used in actual factories.
- **Bias in Surface Materials:** Metal surfaces make up the majority of the dataset. It's possible that the trained model won't generalize well to flaws on non-metallic surfaces (such ceramic or plastic).
- **Bias in annotations:** Subjectivity may be introduced via human labeling. Certain flaws could be hard to distinguish, which could result in incorrect labeling.

The following mitigation techniques can be used to enhance model resilience and address biases:

- **Augmenting data:** To replicate real-world fluctuations, random adjustments including noise addition, brightness adjustment, rotation, and scaling are applied to help in the generalization of models to various angles and lighting situations.
- **Gathering Data in the Real World:** Adding pictures from actual industrial settings to the collection of pictures of the original dataset helps in to representing varying degrees of defect severity.
- **Creation of Synthetic Data:** To create artificial defect images, domain adaptation strategies or GANs (Generative Adversarial Networks) [28] are applied to help in balancing uncommon flaws that might not be well-represented in actual datasets.
- **Transfer Learning models:** Before fine-tuning on the NEU dataset, train models on bigger, more varied datasets such generic industrial defect datasets which helps in keeping small datasets from becoming overfitted.
- **Class Rebalancing:** In deep learning models, using class-weighted loss functions guarantees that faults with higher real-world occurrence are given the proper attention.
- **Multiple-View Inspection:** Train models include photographs taken from various perspectives and lighting conditions enhances flaw detection in real-world applications.

According to the mentioned biases of the NEU-DET dataset, some of mentioned limitations results from images being captured in a controlled setting, leading to risks of overfitting and poor generalization. To address these issues, a multi-faceted strategy was implemented within the experimental pipeline. Data augmentation was performed using Keras' ImageDataGenerator, including random rotations, shifts, flips, and brightness adjustments to simulate industrial environment variability. A two-phase fine-tuning strategy was employed with the pre-trained ResNet50V2, where only the last 50 layers were unfrozen for adjustment to prevent overfitting, alongside a Dropout layer added for regularization. Comprehensive metrics including Precision, Recall, F1, and multi-class ROC-AUC curves were used to ensure balanced assessment and avoid bias from aggregate scores. The techniques implemented specifically address dataset limitations and enhance model robustness in training and evaluation.

3.2. Data Preprocessing

To ensure reproducibility and transparent reporting, all experiments were conducted under the following controlled conditions. The software environment utilizes TensorFlow 2.12.0 with Keras API. The NEU Surface Defect Database was partitioned using stratified sampling into 70% training (1,260 images), 15% validation (270 images), and 15% testing (270 images) subsets, with a fixed random seed of 42 applied to guarantee reproducible splits across all experimental runs. The preprocessing steps are critical for ensuring that the input image works with the trained deep learning model. Each stage of the preparation pipeline is described in detail, step by step, below:

- To prepare image for neural networks, it has to be shrunk to 200x200 pixels to maintain uniformity. This step is crucial as neural networks require set input dimensions. Resizing promotes consistency between pictures, and if trained on grayscale photos, the image may need to be transformed to grayscale instead of RGB. Different scaling approaches, such as nearest neighbor [29] or bilinear interpolation [30], can be employed to reduce distortion.
- The image must be converted into a numerical format for the deep learning model to recognize. This involves transforming the image into an array of pixel values, with each pixel representing the intensity levels of Red, Green, and Blue (RGB). The image is shown as a three-dimensional array with width of 200 pixels, height of 200 pixels, and color channels (3 for RGB and 1 for grayscale). A 200x200 RGB image is represented as an array with the dimensions (200, 200, 3).
- The next step in deep learning is to normalize pixel values, which scales pixel intensity values to a smaller range, usually 0 to 1. This is achieved by dividing all pixel values by 255, as the original values span from 0 to 255. This normalization procedure improves neural networks' performance, prevents numerical instability, and enhances model generalization by making it more resilient to input image fluctuations. However, without normalization, the model may fail to train and generate bad predictions due to large input values. Normalization is crucial for deep learning models using activation functions like ReLU, and not normalizing input images during inference can lead to inaccurate findings.
- To enhance model generalization and address dataset limitations identified in Section 3.1, extensive data augmentation was applied exclusively to the training data. This included random rotations (±30°), width and height shifts (±30%), zoom variations (±40%), horizontal and vertical flipping, and brightness adjustments (60–140% of original intensity). All augmented images utilized nearest-neighbor pixel filling. Validation and test images underwent only normalization without augmentation to ensure unbiased evaluation.
- Deep learning models require input data in batch format. To handle one image, an extra dimension is added to the image, changing its shape from (200, 200, 3) to (1, 200, 200, 3). This new dimension represents a batch size of one, allowing the model to handle the image properly. Without this step, the model may raise errors, especially for Convolutional Neural Networks (CNNs), which are designed to handle numerous pictures simultaneously during training and inference.
- The process of image classification involves sending the processed image to a trained deep learning model for categorization. The model evaluates the image and provides likelihood ratings for each defect category, indicating its confidence in predicting a fault. The class with the highest likelihood is chosen as the projected defect type. For example, if the model produces probabilities for crazing, inclusion, patches, pitted surface, Rolled-in Scale, and scratches, the image is classified as a "Rolled-in Scale" fault due to its 60% likelihood. However, if the probability of numerous classes is comparable, the model may be questionable.

3.3. Feature Extraction

Feature extraction is an important stage in developing a machine learning model, especially for image classification tasks such as defect identification in the NEU Surface Defect Dataset. Feature extraction aims to transform raw image data into significant representations that the model can comprehend and utilize for categorization. CNNs are widely employed in deep learning to extract features from photos.

The NEU Surface Defect dataset comprises grayscale images featuring six unique defect categories: crazing, inclusions, patches, pitted surface, rolled-in scale, and scratches. Each variety exhibits unique patterns, including variations in texture, edge configurations, shape characteristics, and contrasts in contrast. The feature extraction

method seeks to identify these characteristics to allow a deep learning model to differentiate among defect categories. The dataset comprises textural, structural, and shape-based attributes, enabling the model to precisely identify and categorize faults. The image preprocessing process involves several steps to extract features from the data. These include resizing the images to 200x200 pixels for consistency, normalizing the pixel values to ensure training stability, converting the image to a numerical matrix (grayscale pixel values) for feature extraction, and expanding the dimensions to batch format for deep learning algorithms. These steps ensure efficient feature extraction and minimize numerical fluctuations, ensuring a stable and effective learning process.

1. **Input Preprocessing**: Input images were preprocessed to match the model's expected format. Each image was resized to 224×224 pixels and its pixel values were normalized to a range of [0, 1].
2. **Fixed Feature Extractor**: During the first training phase, all layers of the pre-trained ResNet50V2 base model were frozen (`base_model.trainable = False`). This means the model's convolutional layers — which include operations like 7×7 and 3×3 convolutions, batch normalization, and residual connections — acted as a static feature extractor. These layers transformed each input image into a high-dimensional, abstract feature map representing patterns like edges, textures, and shapes learned from ImageNet.
3. **Feature Map Output**: The final output of this frozen base model was a 3D feature map (with dimensions like `(batch_size, 7, 7, 2048)` for ResNet50V2), which contained the extracted hierarchical features. This feature map served as the input to the new, trainable classification head.
4. **Custom Classification Head**: The feature map was then passed through a custom classification head consisting of: a Global Average Pooling 2D layer to reduce spatial dimensions, a Dense(256, ReLU) layer for non-linear combination of features, a Dropout(0.5) layer for regularization, and a final Dense(6, softmax) layer for defect classification.

After preprocessing the image, the features are extracted using the CNN. The CNN retrieves features using numerous layers of processing, each with a greater level of information as illustrated in the following:

A. **Convolutional Layers: Edge and Texture Detection.** The initial stage of feature extraction involves using convolutional layers to detect basic picture patterns. The first layer detects edges and lines using small filters, such as 3x3 or 5x5. As the image progresses through deeper layers, the model detects increasingly complicated elements, such as crossing cracks in crazing flaws or continuous patterns simulating a textured surface in rolled-in scale.
B. **Activation Functions (ReLUs)**: Following each convolutional level, an activation function, such as ReLU (Rectified Linear Unit) [31], is employed to incorporate nonlinearity. This enables the model to identify intricate patterns rather than merely linear edges. In the absence of an activation function, the CNN would function merely as a basic filter, incapable of detecting intricate patterns. The ReLU function ensures that only essential functions are activated.
C. **Pooling Layers (Dimensional Reduction)** Pooling layers minimize feature map size while retaining critical information. Max pooling selects robust feature responses from each region, reducing redundancy while preserving visible edges and textures. For example, a 5x5 feature map can be reduced to a 2x2 map, retaining only the most significant elements. Scratch patterns and pitted surfaces maintain distinct patterns.
D. **Fully Connected Layers (High-Level Feature Representation)** This procedure encompasses convolution and pooling layers to convert extracted feature maps into one-dimensional vectors for classification. The vector is further processed through dense layers for final predictions. The layers categorize problems according to combinations of features. The concluding classification phase employs a Softmax activation function [32] to convert the feature vector into probability scores for each defect category, with the class exhibiting the highest probability designated as the final prediction.

3.4. Model Selection and Architecture Rationale

Two convolutional neural network architectures were strategically selected to explore the accuracy-efficiency trade-off critical for industrial deployment. MobileNetV2 was chosen for its lightweight design utilizing depthwise separable convolutions, making it suitable for resource-constrained or real-time applications. ResNet50V2 was

selected for its deep residual architecture (50 layers) that effectively mitigates vanishing gradient problems, enabling superior feature extraction for maximum classification accuracy. Both models were initialized with ImageNet-pretrained weights to leverage transfer learning. The original classification heads of both pretrained models were removed and replaced with a custom classification head tailored for six-class defect identification. For both architectures, this included a max-pooling layer (3×3 kernel), a dense layer with 64 ReLU-activated units, a dropout layer (rate=0.5), a global average pooling layer, and a final softmax-activated dense layer with six output units.a dropout layer (rate=0.5), a global average pooling layer, and a final softmax-activated dense layer with six output units.

3.5. Custom Classification Layers

The NEU Surface Defect Identification model uses customized classification layers instead of the default ones from the ImageNet dataset [33]. This is necessary as the initial model only requires six fault categories. The additional layers improve defect categorization and enable better generalization to unseen fault images, enhancing the model's effectiveness.

The pre-trained models have fully connected layers for generic object recognition, but these are useless for surface defect classification. To modify the model, the last categorization layers are eliminated and convolutional layers are kept for feature extraction. This allows the model to reuse learned visual elements like edges, textures, and patterns for identifying flaws in steel surfaces.
The feature extraction layers are followed by several layers to classify surface faults. The Global Average Pooling (GAP) layer reduces the feature map's spatial dimensions to one vector, reducing overfitting and improving model robustness. The dense layer, consisting of 256 neurons with a ReLU activation function, learns combinations of extracted information to distinguish fault types and find deeper patterns in faults. A 50% dropout layer mitigates overfitting by randomly deactivating half of the neurons during training, compelling the model to acquire substantial features rather than memorizing noise. The concluding classification layer employs a Softmax activation function [34], with six neurons that represent six defect groups, with the category exhibiting the highest probability selected as the ultimate prediction.
The model is trained in two stages: freezing the pre-trained layers and adjusting the model to accommodate defect features. The model freezes all layers initially, allowing only new classification layers to be trained using the faulty dataset. This prevents overfitting on tiny datasets and allows the model to reuse generic features without affecting pretrained weights. The model is then trained with a low learning rate to modify only specific layers, allowing the model to learn industry-specific features for fault detection.

3.6. Model Training and Fine-Tuning Protocol

A systematic two-phase transfer learning strategy was implemented for both MobileNetV2 [40] and ResNet50V2 [35] models. A two-phase transfer learning strategy was employed to adapt the pre-trained ResNet50V2 model to the steel defect classification task. The base model was initialized with weights pre-trained on the ImageNet dataset. In the first phase, all layers of the base model were frozen, and only the newly appended custom classification head was trained for 10 epochs. This allowed the model to learn task-specific decision boundaries while preserving the generic feature representations from ImageNet. For the second, fine-tuning phase, a strategic layer-unfreezing approach was implemented. The last 50 layers of the ResNet50V2 base model were unfrozen, enabling the model to adapt its higher-level, task-specific feature extractors to the nuances of steel surface defects. To ensure stable weight updates and prevent catastrophic forgetting of useful pre-trained features, training resumed with a significantly reduced learning rate of 1e-5. The model was fine-tuned for an additional 10 epochs using a batch size of 32. This phased approach effectively balanced the retention of valuable pre-trained knowledge with the specialized adaptation required for the target domain.

3.6.1. Training Strategy and Hyperparameters In Phase 1 (feature extraction), all pretrained layers were frozen while only the newly added classification layers were trained using the Adam optimizer with an initial learning rate of 0.001 and categorical cross-entropy loss. Training employed a batch size of 32 for 30 epochs with early

stopping (patience=5) based on validation loss. Exponential learning rate decay (decay factor = 0.9 per epoch) was applied throughout training. In Phase 2 (fine-tuning), the top 50 layers of ResNet50V2 and proportionally equivalent layers in MobileNetV2 were unfrozen. Training resumed with a reduced learning rate of 0.0001 to enable gradual adaptation to defect-specific features. Regularization techniques included dropout (rate=0.5) in classification layers and L2 weight decay ($\lambda = 0.0001$).

3.6.2. MobileNetV2 Implementation

The environmental setup for the model development is done using TensorFlow 2.11.0 and trained on a Kaggle environment equipped with a P100 GPU (16GB VRAM) and 13GB of system RAM. The fine-tuning process was applied to the last 50 layers of ResNet50V2 to adapt high-level features specific to steel defects, while keeping the earlier layers frozen to preserve general feature extraction capabilities learned from ImageNet. Following training, the optimized model achieved an average inference time of 0.22 seconds per image on the same hardware, which is suitable for near real-time quality inspection in many industrial settings. The complete source code, including all preprocessing, training, and evaluation steps, has been made publicly available in a persistent repository on Kaggle and can be accessed via the provided link [36] for full transparency and reproducibility. The lightweight MobileNetV2 architecture, utilizing depthwise separable convolutions, was optimized for efficiency. Table 2 summarizes its training procedure.

Table 2. The Training Procedure for MobileNetV2

Stage	**Action**	**Purpose**
Feature Extraction	Train a new classifier after freezing the MobileNetV2 layers	Prevent overfitting by utilizing pre-trained information
Fine-Tuning	Unfreeze deeper layers and train at a slower pace	Enhanced ability to adapt features to surface flaws
Loss Function	Cross-entropy that is categorical	enhances the performance of classification
Optimizer	Adam	guarantees quick and steady convergence
Learning Rate Decay	dynamically modifies the learning rate	prevents unexpected weight changes and guarantees stability
Regularization	Data Augmentation and Dropout	improves the generalization of the model

Table 3. The Training Procedure for ResNet50V2 model

Stage	**Action**	**Purpose**
Feature Extraction	Train a new classifier after freezing the ResNet50V2 base	Prevent overfitting by utilizing pre-trained information
Fine-Tuning	Unfreeze deeper layers and train at a slower pace	Enhanced ability to adjust to defect patterns
Loss Function	Cross-entropy that is categorical	Improve the classification of many classes
Optimizer	Adam	guarantees consistent and flexible learning
Learning Rate Decay	gradually lowers the rate of learning	keeps unexpected weight changes at bay and enhances stability
Regularization	Data Augmentation and Dropout	improves the generalization of the model

3.6.3. ResNet50V2 Implementation
The deeper ResNet50V2 architecture, with its residual connections, was optimized for accuracy. Table 3 summarizes its training procedure.

3.7. Evaluation and Results

Model performance was assessed using multiple metrics calculated on the held-out test set: accuracy, precision, recall, F1-score (per-class and macro-averaged), Area Under the ROC Curve (AUC).

3.7.1. Evaluation Metrics The most important evaluation metrics used for evaluation are derived from the confusion matrix which consists of four basic characteristics (numbers) that are used to define the measurement metrics of the classifier. These four numbers are:

- **True Positive (TP)**: correctly identified image by the model as it contains defects; represents correct defect detection.
- **True Negative (TN)**: correctly identified image by the model as it does not contain defects; signifies correct rejection.
- **False Positive (FP)**: incorrectly identified image by the model as containing defects; indicates false alarm/over-detection, leading to unnecessary rejection of quality steel.
- **False Negative (FN)**: An image containing defects not identified by the model as such; constitutes missed defect/under-detection, potentially allowing defective steel to pass inspection.

The previously mentioned evaluation metrics depend on the TP, TN, FP, and FN.

- **Accuracy** The proportion of all images that were correctly identified to a certain class images as illustrated in equation(1).

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

- **Precision**: The proportion of positive predictions that are correctly identified. Measures model's reliability when it predicts a defect class. According to equation (2), high precision reflects a low false positive rate, meaning the model is accurate when it predicts a condition.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2}$$

- **Recall**: The proportion of actual positive cases that are correctly identified. Measures model's ability to detect all instances of a defect class. From equation (3), high recall indicates that the model successfully captures a large number of true positive cases, minimizing false negatives.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{3}$$

- **F Measure**: The harmonic mean of precision and recall. Provides a single metric balancing false positives and false negatives for a defect class. From equation (4), it is particularly useful when the dataset has an uneven class distribution or when both false positives and false negatives carry significant consequences.

$$\text{F Score} = \frac{2 * precision * recall}{precision + recall} \tag{4}$$

- **ROC-AUC (Receiver Operating Characteristic - Area Under Curve)**: The probability that a randomly chosen positive instance (defect class) is ranked higher than a randomly chosen negative instance (all other classes). Measures class discrimination ability across all classification thresholds and are derived from equation (5), (6).

$$\text{AUC} = \frac{Recall\ Class2 + Recall\ Class3 + Recall\ Class4 + Recall\ Class5 + Recall\ Class6}{6} \tag{5}$$

The mean of AUC values calculated separately for each defect class in one-vs-rest mode. Evaluates overall multi-class discrimination performance.

$$\text{Macro-Average AUC} = \frac{(AUC\_RS + AUC\_Pa + AUC\_Cr + AUC\_Ps + AUC\_In + AUC\_AUC_{Sc})}{6} \quad (6)$$

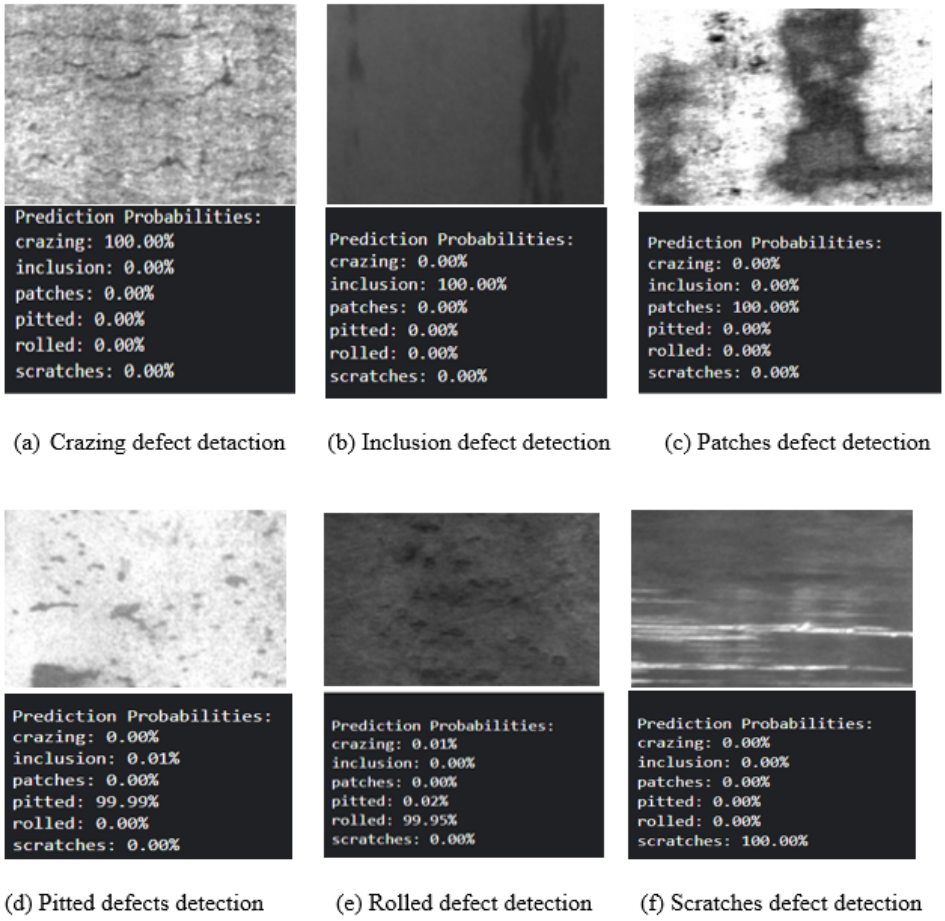


Figure 3. Six tested samples for the six different defects from the NEU-DET dataset and their corresponding detection results by MobileNetV2 model

3.8. Evaluation of MobileNetV2 Model

TThis model has high feature extraction efficiency with low resource consumption, making it ideal for industrial applications. It was created to be both lightweight and effective at classifying images.
The MobileNetV2 model was modified to improve its performance in categorizing metal surface defects in the NEU-DET database. The model was constructed using pre-trained weights on Image Net and was set to untrainable to maintain retrieved features. Personalized layers were added, transforming image features into feature vectors using the GlobalAveragePooling2D layer, and incorporating nonlinear learning using a dense layer. To avoid overfitting, a dropout layer was used. Images were categorized into six classes using a dense layer. The model's training was established using a categorical_crossentropy error function and Adam as the optimizer. The learning rate was gradually reduced using Learning Rate Scheduler. Fine-tuning was used to train the top layers, and the

learning rate was lowered to 0.0001. The model's performance was monitored using validation data during its 30-epoch training, and after multiple epochs, the model's validation accuracy reached 100%, demonstrating a notable improvement. Figure 3 shows the images labeled from (a) through (f) and their corresponding sample outputs from the MobileNetV2 model applied to six distinct steel surface defects from the NEU-DET dataset. Each sub-image displays both the input defect sample and the detection result, indicating how well the model identifies each defect type.

On the other hand, accuracy and loss have been measured in both testing and validation cases in figure 4 for MobileNetV2 model to prove no overfitting detected and higher performance has been achieved.

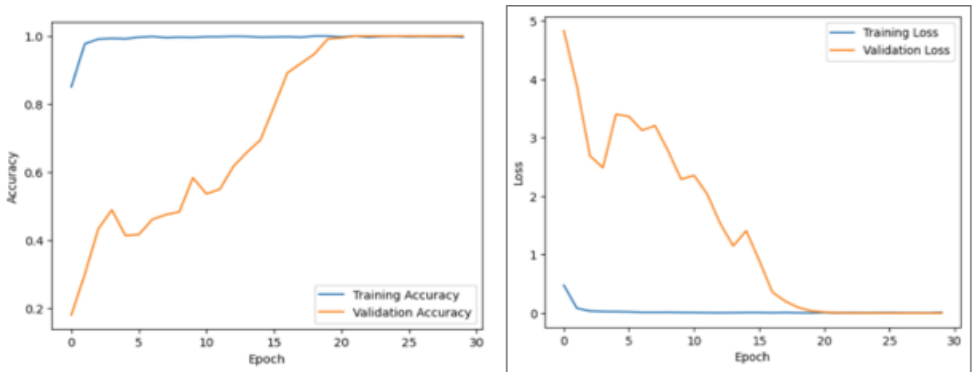
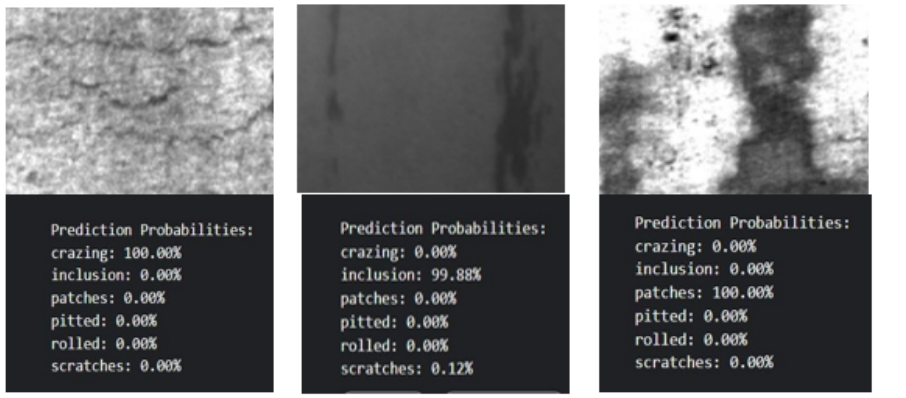


Figure 4. Accuracy and loss measurements in both validation and training cases for MobileNetV2 model on NEU-DET dataset.

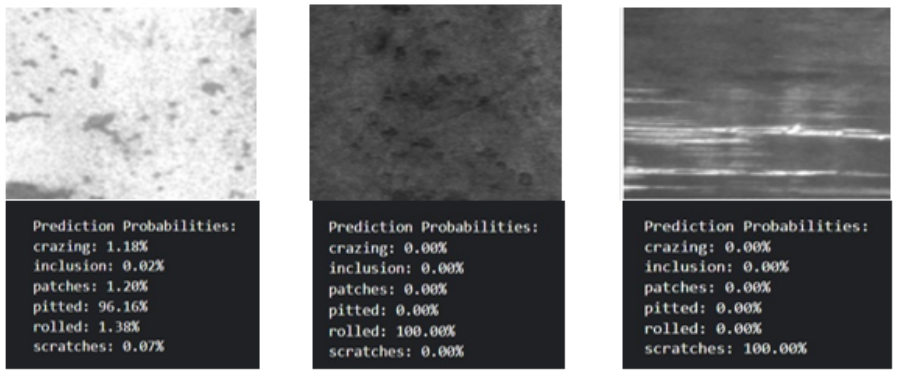
3.9. Evaluation of ResNet50V2 Model

The deep neural network ResNet50V2 serves as the second model utilized in this investigation to categorize surface flaws in the NEU-DET dataset. The NEU-DET database's surface defect detection and classification model is based on the ResNet50V2 architecture, a deep neural network that excels in advanced feature extraction and picture analysis. The model's objective is to identify and categorize surface flaws in the database, which includes photos of surface flaws like holes, cracks, and scratches. The model's elements include preparing data, using the base model of ResNet50V2, including custom layers, training the models, and adjusting the model. The model uses a Learning Rate Scheduler as an optimizer to enhance performance and adjust the weights to make the model more dataset-specific. The model's layout includes the ResNet50V2 Backbone for feature extraction, a Global AveragePooling2D for creating a single vector, a dense layer for learning features, a 0.5 dropout to prevent overfitting, and a dense layer with six classes for the output with the most classes.

The model incorporates tailored layers for surface fault classification while utilizing the representational capability of ResNet50V2. After fine-tuning, the model produced exceptional performance, making it a useful tool for these applications. The model's specifics are described in the text, and the model is saved for future use in future predictions on fresh photos. Figure 5 shows the images labeled from (a) through (f) and their corresponding sample outputs from the ResNet50V2 model applied to six distinct steel surface defects from the NEU-DET dataset. Each sub-image displays both the input defect sample and the detection result, indicating how well the model identifies each defect type.



(a) Crazing defect detaction (b) Inclusion defect detection (c) Patches defect detection



(d) Pitted defects detection (e) Rolled defect detection (f) Scratches defect detection

Figure 5. Six tested samples for the six different defects from the NEU-DET dataset and their corresponding detection results by ResNet50V2 model

Accuracy and loss have been measured in both testing and validation cases in figure 6 for the ResNet50V2 model to prove no overfitting detected and higher performance has been achieved.

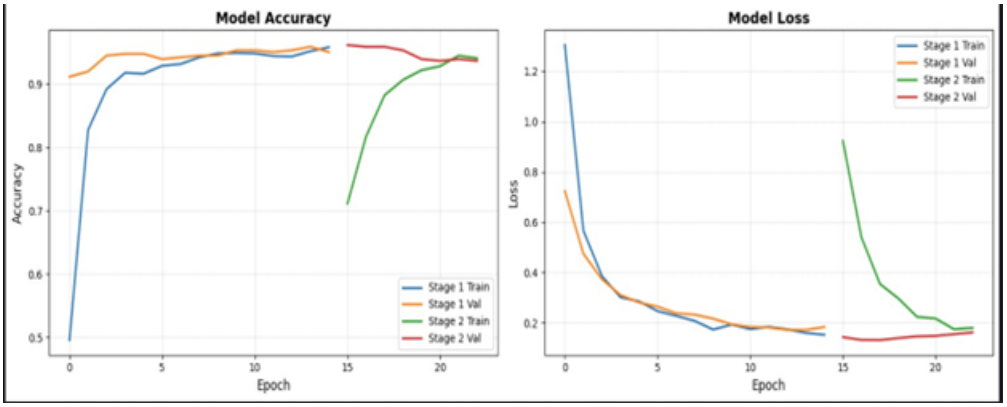


Figure 6. Accuracy and loss measurements in both validation and training cases for ResNet50V2 model on NEU-DET dataset.

The training progression of the ResNet50V2 model is visualized in Figure 7, which plots the Training and Validation AUC scores across epochs. Both metrics show a consistent upward trend, converging at a high plateau of approximately 0.98 after 15 epochs. The minimal gap (¡0.02) between the training and validation curves indicates effective learning without significant overfitting. This demonstrates that the model developed robust feature representations capable of discriminating between defect classes on unseen data, validating the chosen fine-tuning strategy and regularization parameters.

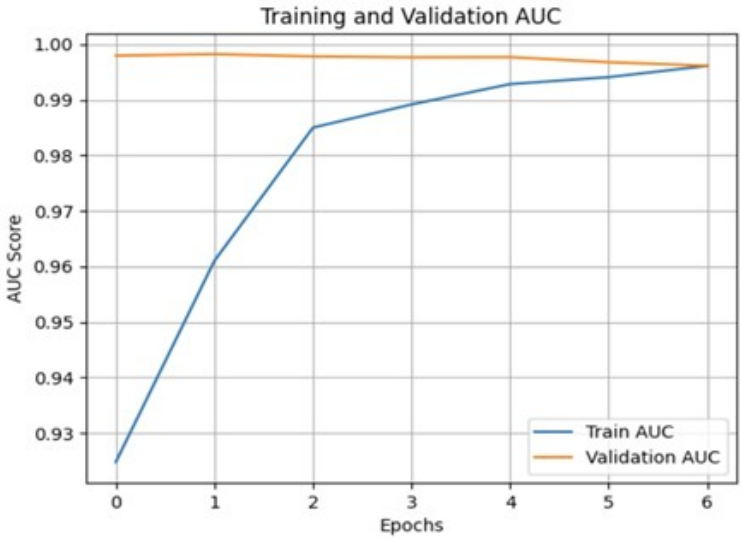


Figure 7. Training and Validation AUC Progression for ResNet50V2 model

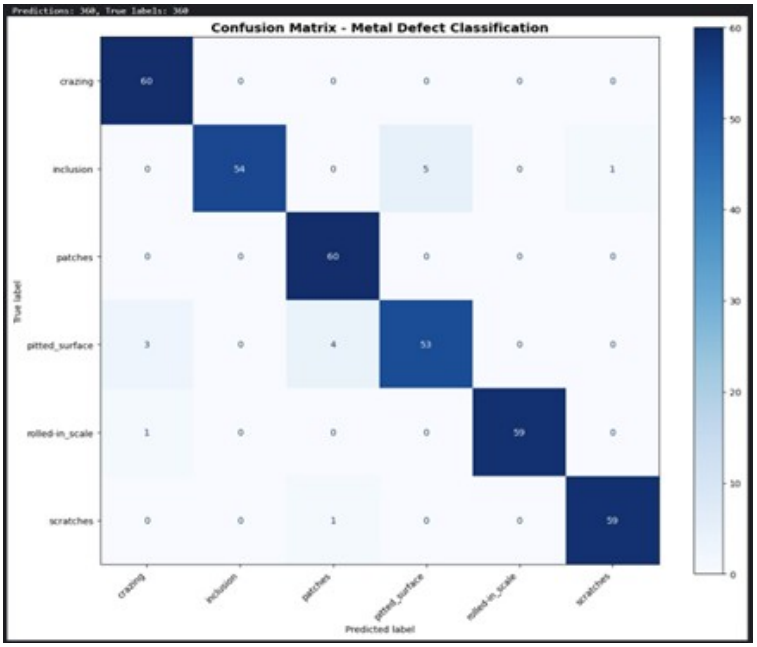


Figure 8. Resnet50 confusion matrix Result after testing on 360 images

Figure 8 presents the confusion matrix for the ResNet50V2 model evaluated on the independent test set. The pronounced diagonal indicates robust multi-class discrimination, with particularly strong performance on scratches and inclusion defects. The primary confusion occurs between crazing and scratches, with 12 mutual misclassifications a pattern likely attributable to similar linear morphological characteristics in the image data. Secondary confusion between pitted surface and patches suggests these defect types share visual similarity in terms of texture distribution. These specific inter-class confusions highlight where feature representation could be enhanced in future work, particularly for distinguishing fine linear defects in steel surfaces.

3.10. Real Test Cases Evaluation and Results

From the previously explained section, it has been conducted that both 2 models have high F1-score of 95%. However, the ResNet50V2 outperforms the MobileNetV2 model, so this model is recommended to be tested in real life or real cases on images outside the range of the NUE-DET dataset. Some pictures of (a) scratched car's paint, (b) metal surface with some crazing,(c) washing machine with patches on surface, and (d) refrigerator with pitted surface. The model succeeded to identify the target defects with high accuracy besides identifying other defects with small percentage in each image as illustrated in figure 9.

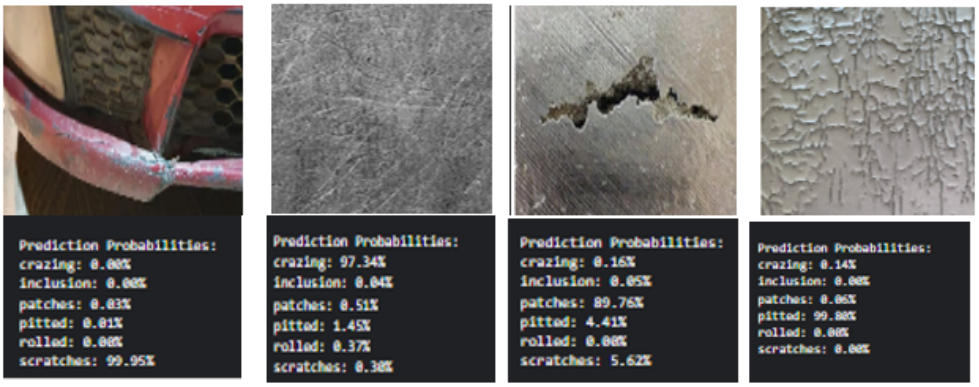


Figure 9. Some pictures of (a) scratched car's paint, (b) metal surface with some crazing ,(c) washing machine with patches on surface , and (d) refrigerator with pitted surface

3.11. Comparative Analysis

The authors in [**?**] implemented the ResNet50 model, as in our work, for defect detection in rolled metal with an accuracy of 96.91%. The model classifies defects into three classes, with an ideal depth of 50 layers for better generalization capabilities. The NEU-DET dataset is used in [32] where it highlights flaw detection weaknesses and suggests a more effective approach. YOLOv10n-SFDC, with enhanced capabilities, achieves 85.5% precision on the NEU-DET dataset, surpassing SSD and Fast R-CNN models. However, it retains less accuracy due to a smaller dataset. The GC10-DET dataset, introduced in [37], addresses defects in metal surfaces using a fault detection network and negative mining technique. The method achieves 98% precision for Cr, In, Pa, Ps, and Rs defects, and outperforms previous models using the NEU-DET dataset. An improvement for YOLOv5 is introduced in [38] by refining its network, feature scales, and detection layer. It integrates convolutional block attention into Inception-ResnetV2, improving second-stage recognition and defect feature extraction. The framework achieves optimal performance with 83.3% precision on the Enriched-NEU-DET dataset. The authors in [39] introduces an improved real-time model, NGD-YOLO, for detecting steel surface defects, enhancing detection accuracy by 4.6% compared to the baseline YOLOv5s network which is 79.2%, utilizing a lightweight NAM and GD mechanism. The EA-YOLO model [40], based on Yolov8, has been presented and case studies with the NEU-DET and PCB-DET datasets show EA-YOLO achieves a mAP of 81.1 and 97.8% A deep learning method [41] using a pre-trained

VGG16 model and a newly developed CNN achieved a classification accuracy of 99.44% on the same used dataset in our work, NEU dataset. A deep-learning-based computer vision techniques are implemented in [42] for detecting surface defects on steel sheets. The U-Net, FCN-8, and FPN models are used for segmentation, while the YOLOv4 model is used for object detection. The U-Net model outperforms existing methods, with a 72% Dice Similarity Coefficient.

Table 4. Comparative analysis of published papers in metal defects detection

Ref.	Model	Dataset	Contributions	Performance Metrics	Results Analysis
51	ResNet50	Defected flat surface images	Classified into 3 defect classes; analysis of model depth	Accuracy: 96.91%	Best depth for ResNet is 50 layers
52	YOLOv10n-SFDC	NEU-DET	Enhanced YOLOv10 with improved capabilities	mAP: 85.5%	Outperforms SSD and Fast R-CNN
53	SSD with systematic negative mining	GC10-DET	Addresses class imbalance via negative mining	Optimal results for Cr, In, Pa, Ps, Rs	Surpasses SSD300 in In and Rs defect detection
54	Improved-YOLOv5 + Inception-ResnetV2 + CBAM	Enriched-NEU-DET	Added CBAM to Inception-ResNetV2; improved base YOLOv5	mAP: 83.3%	Two-stage framework shows superior versatility
55	NGD-YOLO	Steel surface defects dataset	Added NAM attention module, GD mechanism	Accuracy: 79.2%	Reduces information loss in multi-scale fusion
56	EA-YOLO	NEU-DET, PCB-DET	C2f $\rightarrow$ C2FN replacement; environmental awareness network	mAP: 81.1%	High robustness and generalization ability
57	VGG16 + custom CNN	NEU	Combines strong feature extractor with tailored CNN classifier	Accuracy: 99.44%	Handles both diversity and similarity of defects
58	U-Net, FCN-8, FPN + YOLOv4	Steel sheet defects dataset	Two-stage structure for small defects	DSC: 72%	U-Net best for segmentation

4. Conclusion

This work developed a deep learning framework for detecting and classifying steel surface defects using MobileNetV2 and ResNet50V2 models on the NEU Surface Defect Database. The methodology involved standardization, normalization, and augmentation of the dataset, feature extraction using CNN layers, custom classification layers, and fine-tuning of the models. ResNet50V2 demonstrated superior performance in real-world testing, achieving 99-100% accuracy. By applying the highest accurate model on real test cases and measuring accuracy, we obtained accuracy measurements between 89% and 99%. The results show that deep learning can

automate industrial quality control, reduce human error, and improve efficiency. Future work may focus on real-time deployment, edge-compatible models, and synthetic data generation to address dataset biases. This research contributes to advancing AI-driven inspection systems in manufacturing.

5. Future Work and Limitations

While the study shows high accuracy in using deep learning models, especially ResNet50V2, for steel defect detection, significant limitations are present. These include dataset constraints, as the NEU database uses images captured in a controlled environment, lacking variability in lighting, steel surface diversity, and defect severity assessment. Future research aims to enable real-world deployment through a rigorous three-stage plan: First, a quantitative robustness analysis under simulated disturbances like variable lighting and noise. Second, the implementation of domain adaptation techniques (like Adaptive Histogram Equalization and GANs) and the expansion of the dataset to include diverse metal surfaces and finishes. Third, the development and integration of optimized, lightweight models for real-time edge deployment, employing synthetic data generation to mitigate data bias. We will also address the omission of lightweight model comparisons, noting it as a current limitation and proposing future benchmarking studies against models like YOLOv8n or EfficientNet-Lite to evaluate their deployment viability. These combined efforts will create a comprehensive inspection pipeline to enhance the system's reliability, speed, and adaptability in diverse manufacturing settings.

REFERENCES

1. G. Leibfried and N. Breuer, *Point defects in metals I: introduction to the theory*, vol. 81. Springer, 2006.
2. S. I. Platov *et al.*, "Improvement of hot rolling technology to reduce the "rolled-in scale" defect," *Steel in Translation*, vol. 53, no. 4, pp. 291–297, 2023.
3. S. Pawar *et al.*, "Study of white patch defect in automotive grade interstitial free steel," *Journal of Failure Analysis and Prevention*, vol. 20, pp. 1819–1824, 2020.
4. J. H. Park and Y. Kang, "Inclusions in stainless steels—a review," *Steel Research International*, vol. 88, no. 12, p. 1700130, 2017.
5. J. Bai *et al.*, "A comprehensive survey on machine learning driven material defect detection: Challenges, solutions, and future prospects," *arXiv preprint arXiv:2406.07880*, 2024.
6. M. Robotyshyn, M. Sharkadi, and M. Malyar, "Surface defect detection based on deep learning approach," in *IntSol Workshops*, 2021.
7. S. Lee, L.-M. Chang, and M. Skibniewski, "Automated recognition of surface defects using digital color image processing," *Automation in Construction*, vol. 15, no. 4, pp. 540–549, 2006.
8. L. A. O. Martins, F. L. C. Pádua, and P. E. M. Almeida, "Automatic detection of surface defects on rolled steel using computer vision and artificial neural networks," in *IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society*, 2010.
9. S. Abhinav and E. Elakiya, "Metal surface defect detection using deep learning techniques," in *2024 International Conference on Signal Processing, Computation, Electronics, Power and Telecommunication (IConSCEPT)*, 2024.
10. X. Shi *et al.*, "An improved faster r-cnn for steel surface defect detection," in *2022 IEEE 24th International Workshop on Multimedia Signal Processing (MMSP)*, 2022.
11. K. Demir *et al.*, "Automated steel surface defect detection and classification using a new deep learning-based approach," *Neural Computing and Applications*, vol. 35, no. 11, pp. 8389–8406, 2023.
12. R. Khanam *et al.*, "A comprehensive review of convolutional neural networks for defect detection in industrial applications," *IEEE Access*, 2024.
13. H. Zhang *et al.*, "An efficient model for metal surface defect detection based on attention mechanism and multi-scale feature," *The Journal of Supercomputing*, vol. 81, no. 1, p. 40, 2025.
14. Z. Li *et al.*, "A deep learning model for steel surface defect detection," *Complex & Intelligent Systems*, vol. 10, no. 1, pp. 885–897, 2024.
15. Y.-C. Huang, K.-C. Hung, and J.-C. Lin, "Automated machine learning system for defect detection on cylindrical metal surfaces," *Sensors*, vol. 22, no. 24, p. 9783, 2022.
16. X. Sun, J. Gu, S. Tang, and J. Li, "Research progress of visual inspection technology of steel products—a review," *Applied Sciences*, vol. 8, no. 11, p. 2195, 2018.
17. Q. Luo, X. Fang, L. Liu, C. Yang, and Y. Sun, "Automated visual defect detection for flat steel surface: A survey," *IEEE Transactions on Instrumentation and Measurement*, vol. 69, no. 3, pp. 626–644, 2020.
18. H. Shen, S. Li, D. Gu, *et al.*, "Bearing defect inspection based on machine vision," *Measurement*, vol. 45, no. 4, pp. 719–733, 2012.
19. K. Choi, K. Koo, and J. S. Lee, "Development of defect classification algorithm for posco rolling strip surface inspection system," in *2006 SICE-ICASE International Joint Conference*, pp. 2499–2502, 2006.
20. L. M. Xu, Z. Q. Yang, Z. H. Jiang, and Y. Chen, "Light source optimization for automatic visual inspection of piston surface defects," *The International Journal of Advanced Manufacturing Technology*, vol. 91, pp. 2245–2256, 2017.

21. S. Y. Lee, B. A. Tama, S. J. Moon, and S. Lee, "Steel surface defect diagnostics using deep convolutional neural network and class activation map," *Applied Sciences*, vol. 9, no. 24, p. 5449, 2019.
22. D. Soukup and R. Huber-Mörk, "Convolutional neural networks for steel surface defect detection from photometric stereo images," in *International Symposium on Visual Computing*, pp. 668–677, Springer, 2014.
23. S. S. Martínez, C. O. Vázquez, J. G. García, and J. G. Ortega, "Quality inspection of machined metal parts using an image fusion technique," *Measurement*, vol. 111, pp. 374–383, 2017.
24. W. Abbes, J. F. Elleuch, and D. Sellami, "Defect-net: A new cnn model for steel surface defect classification," in *2024 IEEE 12th International Symposium on Signal, Image, Video and Communications (ISIVC)*, 2024.
25. N. J. Chaudhry *et al.*, "Modeling & evaluating the performance of convolutional neural networks for classifying steel surface defects," *arXiv preprint arXiv:2406.14583*, 2024.
26. X. Chen *et al.*, "Hct-det: A high-accuracy end-to-end model for steel defect detection based on hierarchical cnn–transformer features," *Sensors*, vol. 25, no. 5, p. 1333, 2025.
27. F. Riaz, K. Kamal, T. Zafar, and R. Qayyum, "An inspection approach for casting defects detection using image segmentation," in *2017 International Conference on Mechanical, System and Control Engineering (ICMSC)*, pp. 101–105, 2017.
28. F. Constantin and C.-A. Boiangiu, "A review of generative adversarial networks for computer vision tasks," *Electronics*, vol. 13, no. 4, p. 713, 2024.
29. S. Har-Peled *et al.*, "Approximate nearest neighbor: Towards removing the curse of dimensionality," *Theory of Computing*, vol. 8, pp. 321–350, 2012.
30. Y. Lu *et al.*, "Bilinear interpolation algorithm based on verilog," in *2024 IEEE 7th International Conference on Electronic Information and Communication Technology (ICEICT)*, pp. 1375–1379, 2024.
31. I. Konovalenko *et al.*, "Steel surface defect classification using deep residual neural network," *Metals*, vol. 10, no. 6, p. 846, 2020.
32. L. Liao *et al.*, "A novel yolov10-based algorithm for accurate steel surface defect detection," *Sensors*, vol. 25, no. 3, p. 769, 2025.
33. O. Russakovsky *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, pp. 211–252, 2014.
34. R. S *et al.*, "Digital implementation of the softmax activation function and the inverse softmax function," in *2022 4th International Conference on Circuits, Control, Communication and Computing (I4C)*, pp. 64–67, 2022.
35. S. Riyadi, F. A. Abidin, and N. Audita, "Comparison of resnet50v2 and mobilenetv2 models in building architectural style classification," in *2024 International Conference on Intelligent Systems and Computer Vision (ISCV)*, 2024.
36. Kaggle, "Neu surface defect - resnet50v2." https://www.kaggle.com/code/omaramir2001/neu-surface-defect-resnet50v2-96351c, n.d.
37. X. Lv *et al.*, "Deep metallic surface defect detection: The new benchmark and detection network," *Sensors*, vol. 20, no. 6, p. 1562, 2020.
38. Z. Li *et al.*, "A two-stage industrial defect detection framework based on improved-yolov5 and optimized-inception-resnetv2 models," *Applied Sciences*, vol. 12, no. 2, p. 834, 2022.
39. B. Li *et al.*, "Ngd-yolo: An improved real-time steel surface defect detection algorithm," *Electronics*, vol. 14, no. 14, p. 2859, 2025.
40. B. Li *et al.*, "A small defect detection technique for industrial product surfaces based on the ea-yolo model," *The Journal of Supercomputing*, vol. 81, no. 2, p. 415, 2025.
41. A. A. M. S. Ibrahim and J. R. Tapamo, "Transfer learning-based approach using new convolutional neural network classifier for steel surface defects classification," *Scientific African*, vol. 23, p. e02066, 2024.
42. S. Ashrafi *et al.*, "Steel surface defect detection and segmentation using deep neural networks," *Results in Engineering*, vol. 25, p. 103972, 2025.