

Article

Resource-Efficient Surface Defect Detection on Edge Devices Using a Hybrid Descriptor Framework with Data-Driven Feature Selection

Burhan Duman 

Department of Computer Engineering, Isparta University of Applied Sciences, 32040 Isparta, Turkey; burhanduman@isparta.edu.tr

Abstract

Surface defect detection across diverse surface types, such as biological shells, photovoltaic panels, and pavement infrastructure, is essential for industrial quality control and structural health monitoring. Although Deep Learning (DL) models perform well in this domain, high computational and memory requirements limit their deployment on resource-constrained edge devices. To address this, we propose a computationally efficient descriptor-level feature extraction framework combining Local Binary Pattern (LBP), Gabor filters, and Discrete Wavelet Transform (DWT) to capture complementary textural, directional, and frequency characteristics of surface defects. An AdaBoost-driven feature selection strategy reduces the high-dimensional hybrid feature pool to the 25 most discriminative attributes, and SHapley Additive exPlanations (SHAP) analysis is applied to interpret feature contributions. The proposed framework achieved 100% accuracy on the EggCrack dataset and 79.0% accuracy on the ELPV dataset, which is more challenging. On ELPV, it outperformed the MCU-optimized MCUNet-In0 (78.16%) and remained within 4.16 percentage points of EfficientNetV2-B0 (83.16%), while achieving approximately 7577× fewer floating-point operations (FLOPs) and 89× smaller model size. Deployed on a Raspberry Pi 5 CPU, the framework achieved an end-to-end inference time of 37 ms per image, enabling real-time inference. These results indicate that the proposed hybrid descriptor and selection framework offers a favorable accuracy-efficiency trade-off for real-time surface defect detection on CPU-based edge devices.

Keywords: edge intelligence; surface defect detection; hybrid feature fusion; data-driven selection; resource-efficient computing; explainable AI; real-time image processing



Academic Editor: George A. Tsihrintzis

Received: 28 July 2026

Revised: 27 August 2026

Accepted: 30 August 2026

Published: 31 August 2026

Copyright: © 2026 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Surface defect and crack detection play an essential role in numerous applications such as industrial quality control, infrastructure inspection, biomedical imaging, and photovoltaic panel monitoring. Even though a lot of work has been done in this field, it still poses considerable challenges mainly due to the scarcity of labeled data, the tiny and irregular shape of defects, and the complex background textures where it is hard to identify subtle changes on the surface [1]. The mentioned problems are aggravated when looking at edge computing scenarios like handheld inspection devices, drone monitoring systems, and embedded platforms where strict limitations in latency, memory, and power consumption reduce the feasibility of computationally expensive models [1–3].

Automated defect detection methods have been improved mainly through two approaches: traditional manual feature engineering and deep learning-based models [4,5].

Traditional methods involve a lot of efforts in manual feature design, which is usually very time-consuming. They are also sensitive to environmental changes. CNNs have achieved remarkable success by learning representations that capture textural, structural, and shape-related information directly from the data [4,5]. However, the representational capacity of these models often compromises computational efficiency, limiting their applicability in resource-constrained detection scenarios. Some studies have shown that hybrid approaches combining features extracted from pre-trained deep models (e.g., VGG16, ResNet) with traditional classifiers such as Support Vector Machine (SVM) or Extreme Gradient Boosting (XGBoost) can improve classification accuracy and robustness [6–8]. Vision Transformer (ViT)-based models also stand out due to their success in capturing long-range dependencies in images through self-attention mechanisms [9–11]. However, the computational complexity of the attention mechanism in the ViT architecture is even higher than that of CNNs [1,2]. Such approaches still rely on high-capacity backbone networks, which limit their suitability for real-time deployment on CPU-based edge devices. For these reasons, lightweight feature engineering approaches remain a practical choice for edge-focused defect inspection tasks under stringent computational constraints [1,3,4,12].

In this study, a descriptor-level design combining Local Binary Patterns, Gabor filters, and Wavelet transforms is adopted to capture the complementary textural, directional, and multi-scale features of surface defects [4,5,13,14]. Handcrafted descriptors are generally not intended to substitute deep learning-based models, but they still provide a computationally efficient and sustainable solution when only a small amount of training data is available and GPU acceleration is not an option [1–3]. On the other hand, a natural consequence of the combination of several descriptors is the generation of high-dimensional feature vectors. It thus becomes necessary to employ data-driven feature selection and dimension reduction methods to adapt this high-dimensional feature vector to resource-limited edge devices for efficiency enhancement [3,12,15,16]. The suggested design methodology here puts the main emphasis on feature concatenation at the descriptor level, systematic feature selection, and suitability for deployment on CPU-based edge devices. To the best of our knowledge, the triple combination of LBP, Gabor, and DWT features for defect detection on resource-constrained edge devices has not been systematically investigated before. Previous studies have either used these features individually or in pairs or have focused directly on deep learning; an evaluation of the triple combination—whether with or without a selection mechanism—has not been found in the literature.

Motivation and Contributions

The main motivation of this article lies in the high hardware requirements of deep neural networks [17,18] and the representational limitations of single-feature methods [19,20]. The study, in an attempt to address these weaknesses, offers a full framework in which a hybrid pool of complementary handcrafted features (LBP, Gabor, and Wavelet) is first optimized by data-driven methods, and then the selected features are classified using lightweight classical machine learning algorithms. The proposed solution removes the reliance on computationally heavy GPU-accelerated architectures, thus significantly lowering the computational burden and memory requirement of CPU-based edge devices. Hence, the proposed explainable framework demonstrates an optimal balance between classification accuracy and computational efficiency can outperform state-of-the-art deep learning models on datasets from different domains like food, energy, and infrastructure. Below are the main contributions of this work to overcoming the limitations outlined earlier:

- This study introduces a method for descriptor-level feature extraction that combines handcrafted descriptors (LBP, Gabor, and Wavelet) in a systematically complementary manner. The architecture is intended to extract the textural, directional, and multi-scale

features of defects for CPU-based edge devices, thus offering a viable alternative to deep learning methods for small-to-medium-sized surface defect datasets with limited variability and constrained computational resources.

- The study thoroughly explores how data-driven feature selection influences both classification accuracy and computational efficiency. Filter-based (ANOVA f-classif) and model-embedded (AdaBoost importance) methods are measured against projection-based dimensionality reduction approaches.
- This study employs SHAP to enhance model transparency by quantifying the contributions of selected handcrafted features to the classification decisions, thereby providing feature-level insights into relevant texture, orientation, and frequency characteristics.
- The suggested method has been experimentally validated by addressing three different defect domains, such as biological (EggCrack), industrial (ELPV), and infrastructure (PaveCrack) datasets. Robust accuracy across different domains proves that the proposed method is not limited to a single dataset and is capable of generalization.
- This study provides a comparative evaluation of the suggested method against lightweight CNN architectures (MCUNet, EfficientNetV2B0, and MobileNetV2) in terms of accuracy, computational efficiency, and inference time. This contribution shows the feasibility of achieving competitive accuracy at significantly lower inference time and training costs, highlighting the suitability of descriptor-based methods for deployment on CPU-based edge devices.

The rest of this study is organized as follows: Section 2 surveys related work and the current state of lightweight models; Section 3 describes the proposed methodology; Section 4 shows the experimental results and discussion; Section 5 concludes the study.

2. Related Work

The problem of real-time (automatic) surface defect detection has been addressed in the current literature through four main approaches: (i) classical image processing techniques, (ii) feature engineering and classical machine learning, (iii) lightweight deep neural networks, and (iv) vision transformers.

2.1. Image Processing-Based Approaches

Image processing-based approaches detect surface defects by analyzing deviations in patterns and intensity relative to the background. Traditional systems utilize fundamental techniques ranging from edge detection (e.g., Canny, Sobel) and thresholding to frequency analysis like wavelet transforms. Research has established that these methods may produce very good outcomes in certain application areas [21–23]. One example is the application of thresholding and morphological operator combinations in the classification of cracks on asphalt surfaces [24] and defect detection in MEMS [25] as well as the detection of glass bottle bases [26]. Moreover, these methods have periodically been tweaked to fit the requirements of low-resource hardware like microcontroller-based systems with fuzzy logic [27] and Field Programmable Gate Array (FPGA) solutions for rapid edge detection [28–30]. But these approaches have trouble when they are used in complicated situations that might have various challenges. For instance, these methods are mainly designed for clean and smooth surfaces, and their efficiency usually goes down drastically with the introduction of noise, varying illumination, or even the presence of a heterogeneous background as in outdoor scenes [31–33]. Some of the problems are that the techniques are not able to identify even very fine, low-level cracks or to distinguish a foreign object from a defect [24]. As a result, image processing with traditional methods is increasingly considered a technique to be used in conjunction with more advanced artificial intelligence methods and not as a completely standalone one in the very latest literature.

2.2. Hybrid Approaches with Feature Engineering and Classical Machine Learning

The high computational and energy demands of deep learning pose a sustainability challenge for resource-constrained edge devices, driving the shift toward “Green AI”. Hybrid approaches, which combine hand-crafted features with classical classifiers (e.g., SVM, Random Forest, XGBoost), offer a resource-efficient alternative. Literature indicates that incorporating domain knowledge into feature extraction can rival deep neural networks, particularly when data is limited [34]. Several studies confirm the value of feature fusion; for instance, Deng et al. [13] showed that combining Scale-invariant Feature Transform (SIFT) and HOG descriptors improves SVM accuracy compared to single features. Similarly, the combination of HOG and LBP for detecting tire defects reached 84% accuracy in only 0.82 s [35]. For bearing fault diagnosis, a hybrid framework using LBP and HOG with stacked ensemble classifiers (XGBoost, Random Forest, and Extra Trees) achieved 95% test accuracy [36]. Meanwhile, [37] achieved an 88% F1-score on steel surfaces using wavelet thresholding and Complementary Local Binary Pattern (CLBP) with KNN and Random Forest (RF) algorithms. More recently, a study [38] combined MSLBP, DTCWT, and Gabor Wavelet with NCA feature selection and ensemble AdaBoost for steel defect detection, achieving 98.8% accuracy. Still, concatenation needs thorough testing; Ghilardi et al. [39] found that although LBP alone got 96.6% accuracy on satellite images, the addition of Gray Level Co-occurrence Matrix (GLCM) actually worsened the results to 90.9%.

Hardware-specific implementations additionally substantiate this line of thinking. For instance, an offline LSTM model augmented with handcrafted features (Gabor, LBP, HOG, and GLCM) demonstrated robust fault recognition on CCTV sewer inspection data [40]. Pemila et al. [41] introduced a real-time vehicle classification system on an FPGA (myRIO) platform leveraging random wavelet transforms and XGBoost. This hybrid system attained 98.81% accuracy, exceeding the results of traditional methods while optimizing memory usage and processing time under varying environmental conditions.

Feature extraction generates high-dimensional vectors, which, if not managed, can lead to a problem known as “the curse of dimensionality”. Various reduction techniques such as Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), t-Distributed Stochastic Neighbour Embedding (t-SNE), and Independent Component Analysis (ICA) are therefore used to handle this problem [42]. In their research, Anjum et al. [43] achieved 94% cancer detection accuracy by applying PCA to the combined HOG and Canny feature pool. Also, Liu et al. [44] applied PCA on ISCM, LBP, and GLCM features of steel wire ropes and obtained 97.5% accuracy with a 5-s run time.

Apart from reducing the number of features, feature selection is equally important in improving the performance and the stability of the model, as proven by [45,46]. Accurate selection thus leads to efficient use of resources [47] and shorter model training time [48]. Silva Jr. et al. [49] applied a correlation-based selection method on their wearable ARM Cortex-M devices, which resulted in a 92% decrease in memory usage and 99% reduction in computation time.

In addition, there has been success in improving the accuracy of prediction using strategies such as reliability of features and exclusion of outliers [50–52]. Collectively, these findings have shown that effective feature engineering is a promising solution for edge devices instead of deep learning, which is slow and computationally expensive.

2.3. Lightweight CNNs

CNNs are commonly used for defect detection, yet they tend to be computationally intensive to the point that they may not be run on edge devices [53,54]. In order to make them compatible with low-spec hardware, researchers have focused on efficient architectures and optimization. Yang et al. [55] combined lightweight modules with self-

attention, while Zhang et al. [56] created CrackScopeNet, achieving 82.12% mIoU with only 1.05M parameters and 1.58 GFLOPs. The selection of hardware is also a critical factor; Moosmann et al. [57] demonstrated the RISC-V-based GAP9 processor to be twice as fast and 20% more energy efficient than the MAX78000 for ultra-lightweight architectures.

Optimization techniques such as pruning and quantization play a leading role in the case of microcontrollers. Hong et al. [58] compressed MobileNetv2 to decrease RAM usage by 57% and Flash by 62%. The same goes for Kwon and Park [59], who, with partial weight updates, attained a 46.9% increase in efficiency, and Orășan et al. [60] employed information distillation to develop compact models. At the software level, Novac et al. [61] doubled energy efficiency by using 8-bit/16-bit integers through MicroAI, whereas Macia-Lillo et al. [62] optimized the inference speeds of the ARM Cortex-M.

Despite these improvements, even lightweight CNNs inherently rely on complicated matrix multiplication tasks. In addition to compromising the accuracy of the models, methods such as pruning and compression often impose significant limitations on the model and involve complex retraining processes and hardware, such as specific instruction sets. Additionally, typical microcontrollers rarely come with the hardware accelerator (NPU/GPU) required to achieve real-time speeds.

2.4. Vision Transformers

Transformers are great at capturing dependencies, but the main hurdle is still the fact that they require so much processing power [9]. Nevertheless, more recent architectures such as Swin-T and DeiT-S are nearly as efficient as hardware [10,11]. In the field of industrial applications, there have been various adaptations such as PC-DETR, which is capable of running at 40 Frames Per Second (FPS) for road monitoring [63], and CrackTinyNet (CrTNet), which has achieved 31.3 FPS with the enhanced small-object detection [64].

Drastically cutting down the number of parameters is the main focus for lightweight designs trying to bypass hardware limitations. LW-ViT managed to reduce the number of parameters by 53.8% compared with MobileViT while the accuracy was 95.8% [65]. In the same manner, LW-Transformer was able to achieve an excellent result and accomplished it with only 24M parameters [66], whereas the sparse SRDD model was able to perform at 16.56 FPS on UAVs equipped with 19M parameters [67]. Furthermore, compression techniques are driving IoT adoption; Melikyan et al. [68] halved latency using knowledge distillation, and Yang et al. [69] reduced weight storage by 66.1% via pruning. Hardware innovations, such as compute-in-memory [70] and dynamic quantization for wearables [71], continue to enable real-time analysis on edge devices.

Although lightweight ViTs reduce the number of parameters compared to classical ViTs, their memory usage and the complexity of their attention mechanism remain significantly higher than feature-based methods, limiting their use for edge devices. Furthermore, they cannot be used effectively in low-data scenarios, which are very common in industrial applications.

2.5. Summary and Research Gap

The literature review in the above subsections reveals a clear balance between different approaches. Although deep learning models (Sections 2.3 and 2.4) generally achieve high predictive performance, their practical deployment is severely limited by the hardware constraints of edge devices and their inherent black-box nature, which lacks transparency for critical industrial applications. On the other hand, traditional image processing techniques (Section 2.1) and simple hybrid models (Section 2.2) are often not robust enough to distinguish between various datasets and complex backgrounds. There is still a gap in the research field for a unified framework that combines the representational power

of feature fusion with the efficiency of classical classifiers and can operate without GPU acceleration. Furthermore, there is a fundamental need to go beyond random feature selection and standard dimension reduction methods to find the most discriminative features for fine-grained surface defects. Additionally, the existing literature appears to overlook the issue of cross-domain generalizability for an efficient, single architecture capable of processing different materials, from biological samples to infrastructure. This work directly addresses these overlooked challenges by presenting a hybrid framework at the descriptor level, discussed in the following section.

The proposed explainable framework, positioned optimally between traditional and deep architectures (Table 1), fundamentally bridges the gap between high-performance vision and applied edge intelligence. It not only delivers real-time computational efficiency on GPU-less edge devices but also successfully outperforms state-of-the-art lightweight deep models in terms of accuracy on challenging datasets, demonstrating that high efficiency does not necessarily require a compromise in detection capability.

Table 1. Comparison of defect detection strategies in terms of computational efficiency and edge device compatibility.

Method Category	Representative References	Typical Accuracy	Computational Load	Edge Compatibility	Key Limitation/Characteristic
Heavy CNNs	He et al. [72]; Seol et al. [73]	High (95–98%)	High (GPU-dependent)	Low (High latency on CPU)	Strong dependence on large labeled datasets and high-end hardware
Lightweight CNNs	Liu et al. [74]; Howard et al. [75]	Moderate–High (90–95%)	Moderate (Optimized inference)	Medium (FPS degradation on CPU)	Limited ability to capture fine-grained morphological details
Vision Transformers	Dosovitskiy et al. [9]; Liu et al. [11]	Very High (96–99%)	Very High ($O(n^2)$, 4.5–55 GFLOPs)	Very Low (Infeasible for MCUs)	Massive parameter size and slow inference speed without acceleration
Hybrid Handcrafted	Anjum et al. [43]; Pemila et al. [41]	Competitive (Dataset- dependent)	Very Low (CPU-friendly)	High (Suitable for MCUs/FPGA)	Sensitivity to noise/illumination; lacks adaptive feature selection
Proposed Method (Multi-Descriptor + Selection)	This Work	High (Competitive to DL)	Very Low (Integer-based, low-latency)	Very High (Optimized for edge memory constraints)	Balances accuracy & efficiency via data-driven selection (No GPU needed)

3. Materials and Methods

3.1. Datasets

To validate the texture-invariant performance of the proposed methodology and its applicability to different industrial scenarios, three datasets with distinct characteristics were used. Some sample images from the datasets are provided in Figure 1.

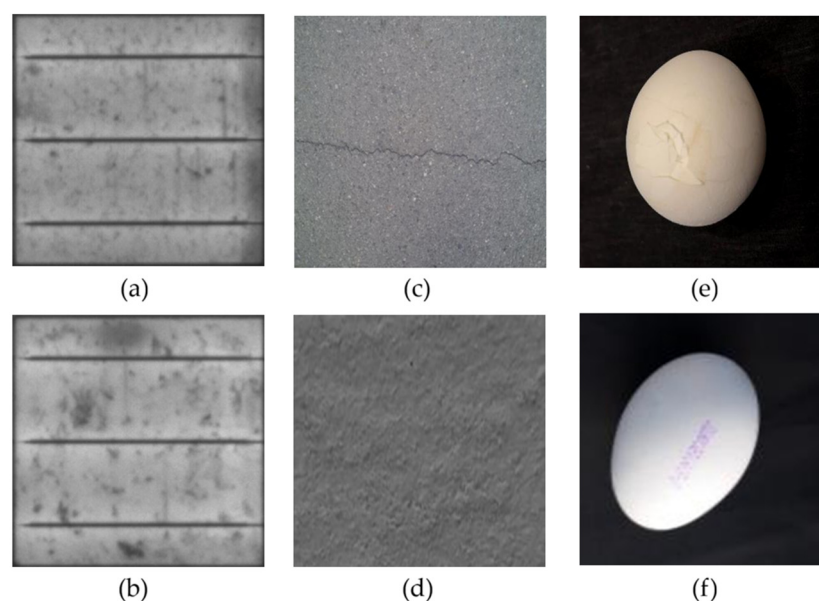


Figure 1. Images of defective and non-defective classes taken from the datasets used in the study: (a) Defect and (b) nodefect images from the ELPV dataset, (c) crack and (d) nocrack images from the PaveCrack dataset, and (e) crack and (f) nocrack images from the EggCrack dataset are presented.

3.1.1. EggCrack Dataset (Biological Surface)

Representing food safety applications, this dataset consists of 2429 egg images classified as cracked, with image dimensions of 640×640 and 640×426 pixels; and 1402 undamaged egg images with a pixel size distribution of 960×720 and 640×640 pixels. The dataset has a curved surface geometry and a homogeneous background image. The cracked egg images were obtained from the Egg crack detection dataset source [76], while the undamaged egg images were obtained using a Samsung SM-M317F mobile phone (Samsung Electronics Co., Ltd., Suwon, Republic of Korea) and data augmentation was performed.

3.1.2. PaveCrack Dataset (Infrastructure Surface)

Representing outdoor concrete asphalt defects, this dataset includes structural damage under the influence of environmental factors such as rough surface texture and shadows. The PaveCrack dataset is divided into two separate classes: “nocrack” and “crack”. Images containing crack defects were collected from the Roboflow platform [77], while images belonging to the “nocrack” class were collected from the dataset shared by the Nigerian Army University on the Kaggle platform [78]. There are 1608 images of the size 227×177 pixels in the “nocrack” category and 1609 images of the size 640×640 pixels in the “crack” category.

3.1.3. ELPV Dataset (Industrial Surface)

This dataset [79], with high-resolution electroluminescence images of photovoltaic cells (ELPV), includes 2624 8-bit grayscale images of monocrystalline and polycrystalline panels at a resolution of 300×300 pixels. The state of healthy and defective solar cells of different damage levels is indicated by 4 different probability values besides the solar module type (mono or polycrystalline). The probability of no damage is 0.0, the probability of minor damage is 0.33, the probability of moderate damage is 0.66, and the probability of severe damage is 1.0.

For the purpose of the study, images with a probability value of 0.0 were assigned to the “nodefect” category, whereas the images with values 0.33, 0.66, and 1.0 were merged into the “defect” class. This conservative binarization ensures that even minor defects (0.33) are not overlooked in a safety-critical application. The crystalline background texture and

high noise level images in the dataset provide an excellent fit for testing the signal-to-noise discrimination capability of the proposed method.

3.2. Proposed Framework

In this section, we propose an intelligent and computationally efficient framework for explainable surface defect detection across various material domains. Specifically designed to overcome the severe hardware constraints of edge environments, the proposed architecture eliminates the reliance on high computational power and large memory capacities. Instead of energy-intensive deep neural networks, it leverages a data-driven, multi-feature extraction strategy that can be deployed seamlessly on standard edge devices equipped with only a CPU. The overall system design, which systematically incorporates hybrid feature fusion, AdaBoost-based selection, and explainable classification stages, is shown in Figure 2.

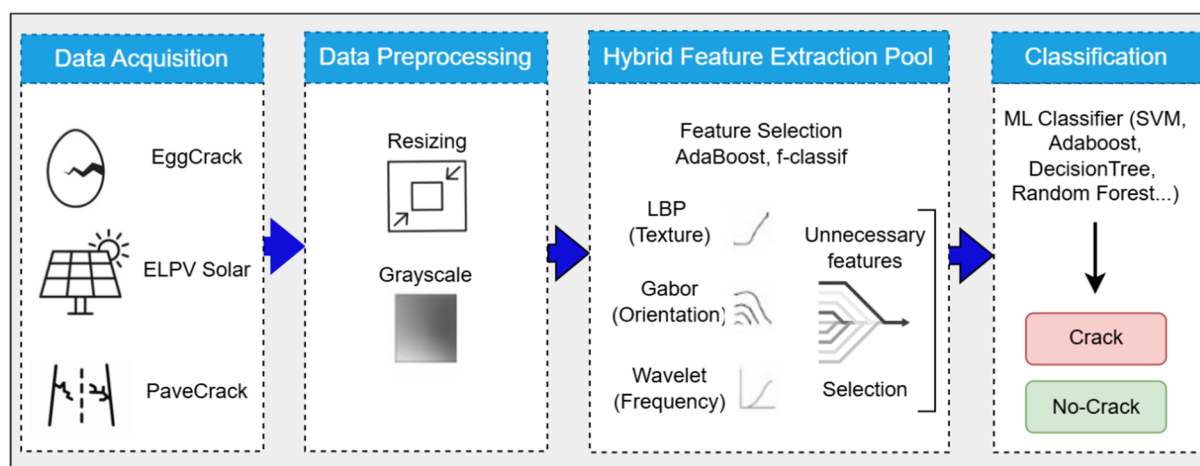


Figure 2. Proposed system architecture.

The system is based on the fundamental principle of combining features extracted using complementary feature descriptors and selecting the most effective ones to understand the structure of complex surface defects of different types. Accordingly, the process consists of data collection, preprocessing, hybrid feature extraction, data-driven feature selection, and classification stages.

To ensure the generalizability of the proposed methodology, images of surface defects in biological (egg crack), industrial (PV panel), and infrastructure (pavement) domains were collected. Each dataset used in the study was obtained from online data repositories (Kaggle, Roboflow, etc.), and some were also obtained from images manually captured with a camera. To reduce images of different sizes in the dataset to a single size and standardize the processing load, color images were converted to grayscale and resized (128×128). A rich feature set was obtained from the preprocessed images by extracting texture information using LBP, orientation and edge information using Gabor filters, and frequency information using wavelet transformation. The extracted features were reduced in number and memory usage by selecting those that best represent the defect using model-based (AdaBoost) and statistical-based (f-class) algorithms. The selected features were used for binary classification using classical machine learning algorithms (SVM, AdaBoost, etc.) to detect “cracks” and “nocracks”. The pseudo-code summarizing the entire pipeline, from raw image input to the specific defect prediction, is presented in Algorithm 1.

Algorithm 1. Data-Driven Hybrid Feature Extraction and Classification Framework.

Input: Raw image I , target resolution (128,128), Selection indices Ω , Trained Classifier Φ
Output: Final Prediction y_{pred} (Defect/NoDefect)
 $I_{resized} \leftarrow$ Resize I to (128,128)
 $G \leftarrow$ Convert $I_{resized}$ to grayscale
Phase 1–3: Hybrid Feature Extraction
 $V_{LBP} \leftarrow$ Extract LBP Histogram from G
 $V_{Gabor} \leftarrow$ Extract Mean/Variance from Gabor responses
 $V_{DWT} \leftarrow$ Extract Energy/Entropy from Wavelet sub-bands
Phase 4: Fusion & Selection
 $V_{total} \leftarrow$ Concatenate $[V_{LBP}, V_{Gabor}, V_{DWT}]$
 $V_{opt} \leftarrow$ Apply selection mask Ω to V_{total} (Reduces dimension to $d = 25$)
Phase 5: Classification
 $Prediction \leftarrow$ Apply Classifier Φ on V_{opt}
If $Prediction$ indicates anomaly **then**
 $y_{pred} \leftarrow$ “Defect”
else
 $y_{pred} \leftarrow$ “NoDefect”
Return y_{pred}

3.2.1. Data Preprocessing

The actual surface defect images that make up the dataset may have different resolutions, varying lighting conditions, and noise. Image size, resolution, and color information also affect the number of features extracted, resulting in more features being extracted. This situation also leads to increased memory usage and computational costs. For the detection of crack defects on the surface, texture and structural change information is more important than color information. While color images are typically three-channel and contain 24-bit data per pixel, grayscale images are represented by one channel of 8-bit data, so grayscale images will require less hardware cost. Furthermore, datasets collected from different sources are likely to have different sizes and resolutions. Deep learning models operate on input images with a resolution of 224×224 and above. High-resolution images may cause memory shortages for edge devices with microcontrollers. For a system designed to run on edge devices with low resource consumption, all images were scaled to 128×128 pixels using the bicubic interpolation method and converted to grayscale.

3.2.2. Hybrid Feature Extraction Pool

Surface crack defects can have very different and complex characterizations in terms of shape, texture, and orientation. A single descriptor/feature extractor may have low performance for defect detection with complex characterization. Therefore, this study aims to obtain a feature set that represents surface defects in a multidimensional way by using complementary LBP, Gabor, and Wavelet methods together.

Local Binary Patterns Descriptor

The LBP operator was used to extract pixel information from the images in order to capture small patterns and lighting changes in the surface texture. LBP also stands out due to its low computational cost. In the LBP operator, the image is divided into regions/blocks of a specific size where each pixel is coded as binary, and the grayscale of the neighboring pixels is examined relative to the center pixel. The tone value of the center pixel is compared

to the value of the neighboring pixel and assigned a binary code (0/1). The neighborhood relationship of each pixel is coded as follows:

$$LBP_{P,R} = \sum_{i=0}^{P-1} S(g_i - g_c)2^i, \quad S(x) = \begin{cases} 1, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases} \quad (1)$$

where g_c and g_i denote the value of the i -th neighboring pixel sampled in the circular neighborhood with radius R around the center pixel and in the P -neighborhood, respectively. Classic LBP produces a relatively high-dimensional feature. Considering resource-constrained devices, multi-scale Uniform LBP has been preferred to represent basic texture structures, increase noise resistance, and reduce the feature vector dimension. Uniform LBP [80] represents transitions from 1 to 0 or from 0 to 1, with a maximum of two transitions across the binary pattern, and is defined by the following function:

$$U(LBP_{P,R}) = |S(g_{P-1} - g_c) - S(g_0 - g_c)| + \sum_{i=1}^{P-1} |S(g_i - g_c) - S(g_{i-1} - g_c)| \quad (2)$$

In a uniform pattern, the number of transitions U should be at most 2. This situation generates only $P + 2$ distinctive features instead of 2^P .

Since surface cracks vary from very small structures to wide cracks, a single radius value may be insufficient to capture all defects. Therefore, the study adopts a multi-scale structure that extracts features at three different scales ($P = 8, R = 1$; $P = 16, R = 2$; $P = 24, R = 3$). By combining the features obtained from each scale, a total of 54 dimensions ($10 + 18 + 26$) of an enriched LBP feature vector representing the surface texture at multiple scales was obtained.

Gabor Filtering

Surface cracks and structural defects typically exhibit a linear characteristic with a specific orientation (directional). Gabor filters, which model the human visual system's directional and frequency-based perception approach, can efficiently characterize textural features for classification. Gabor filters, used to extract the characteristic features of such direction-sensitive structures, are band-pass filters that successfully capture frequency selectivity and spatial locality.

A two-dimensional Gabor filter can be expressed as a sinusoidal wave modulated by a Gaussian function, as follows:

$$g(x, y, \lambda, \theta, \psi, \sigma, \gamma) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right) \exp\left(i\left(2\pi \frac{x'}{\lambda} + \psi\right)\right) \quad (3)$$

$$x' = x \cos \theta + y \sin \theta, \quad y' = -x \sin \theta + y \cos \theta \quad (4)$$

where λ represents the wavelength, θ represents the filter direction, ψ represents the phase shift, σ represents the standard deviation of the Gaussian function, and γ represents the spatial aspect ratio.

The generation of Gabor filters for all possible orientations and scales increases the number of filter banks due to transformations such as rotation and scaling. This results in a high-dimensional feature vector and, consequently, high computational cost, which is a limitation for resource-constrained devices. However, computational cost can be reduced using feature selection or dimensionality reduction methods [81].

In this study, considering computational cost, a Gabor filter bank was generated using four fundamental orientations, $\theta_E \{0, \pi/4, \pi/2, 3\pi/4\}$, sufficient to detect crack orientations, and two different wavelengths (λ and 2λ) to detect cracks of different depths and widths.

The image was convolved with the Gabor kernel in each orientation, and the directional feature vector was obtained by calculating the mean (μ) and standard deviation (σ) values of

each filter response. With this configuration, a total of 16-dimensional ($8 \text{ kernel} \times 2 \text{ statistic}$) directional feature vectors were obtained with 8 Gabor kernels and 2 statistical values. Thus, directional information related to crack characterization is preserved, computational cost is reduced, and edge device compatibility is also ensured.

Discrete Wavelet Transform

Surface defects such as cracks appear as high-frequency distributions embedded in a relatively smooth background. By revealing these characteristics in the frequency domain, fine defects can be separated from the surface texture. The Discrete Wavelet Transform (DWT) was applied for the frequency analysis of the surface image at multiple resolutions [82].

Low-pass and high-pass filters were applied to the image, resulting in a total of 10 sub-bands after three levels of decomposition. One approximation band (cA_3) represents the low-frequency main structure of the image, while 9 detail bands ($cH_i, cV_i, cD_i; i = 1, 2, 3$) represent the horizontal, vertical, and diagonal high-frequency crack information. The Haar wavelet (*Daubechies-1, db1*) was chosen as the main wavelet function due to its success in capturing sharp transitions and its computational efficiency.

The presence of cracks may induce deviations from Gaussianity and heavy-tailed behavior in the distribution of wavelet coefficients. To capture these statistical characteristics, five statistical descriptors—Mean (μ), Standard Deviation (σ), Mean Absolute Deviation, Skewness, and Kurtosis—were computed for each of the 10 subbands to characterize the statistical properties of the coefficient distributions. Consequently, a 50-dimensional feature vector ($10 \text{ subbands} \times 5 \text{ statistics}$), denoted as F_{Wavelet} , was constructed to represent the frequency-domain characteristics of the image.

3.2.3. Feature Fusion Strategy

Point-based methods such as Speeded-up Robust Feature (SURF) and SIFT, which are frequently used in the literature for object recognition, are also used. However, these methods were excluded from the scope of this study because they do not match the characteristics of crack detection and require intensive moving point processing. Furthermore, although the HOG descriptor provided relatively high accuracy in preliminary experiments, it produces a high-dimensional feature vector ($d = 8100$). Therefore, considering resource efficiency, it was not included in the fusion vector.

The proposed triple-feature pool is theoretically motivated by the distinct mathematical domains in which each descriptor operates. LBP is a non-linear operator that encodes local intensity-order relationships directly in the spatial domain, making it sensitive to micro-textural patterns while remaining invariant to monotonic illumination changes. In contrast, Gabor filters are linear band-pass filters defined as Gaussian-modulated sinusoids; they perform localized analysis in the spatial-frequency domain, jointly capturing orientation and frequency information, which is essential for detecting directional edge structures (e.g., elongated cracks). DWT, based on the Haar wavelet, provides a multi-resolution decomposition in the scale-space domain, separating coarse background variations from high-frequency anomalies through approximation and detail sub-bands. Since these three descriptors characterize fundamentally different aspects of image structure—local intensity-order patterns, directional spatial-frequency responses, and multi-scale energy distributions—their integration is mathematically well-founded and complementary. This theoretical rationale is further supported empirically by the low cross-group correlations heatmap and the ablation study, where the triple fusion consistently outperforms all pairwise combinations.

Based on this theoretical complementarity, we adopt a descriptor-level feature fusion strategy to obtain a rich hybrid representation. A rich hybrid feature vector is obtained by

combining the normalized uniform LBP descriptor, Gabor statistical direction features, and wavelet-based frequency statistics:

$$F_{Hybrid} = [F_{LBP} || F_{Gabor} || F_{Wavelet}] \quad (5)$$

The combination method is designed to be simple and deterministic, taking computational cost into account. Feature-level combination provides a rich and explanatory feature space suitable for data-driven optimizations. Less relevant or redundant features in the obtained feature space are removed using a feature selection method to create an edge device-compatible representation. The proposed LBP, Gabor, and Wavelet combination provides the optimal architecture for edge devices due to its suitability for integer-based operations and low memory footprint.

3.2.4. Data-Driven Feature Optimization

The generation of a hybrid feature pool also implies obtaining a large number of features. This situation brings with it high dimensionality and computational cost. The contribution of some of these numerous features to defect detection may contribute very little to defect detection. In this study, “Transformation-based” and “Selection-based” approaches were comparatively analyzed to optimize the obtained hybrid feature vector (F_{Hybrid} -rich feature pool) for edge devices.

Transformation-based methods, such as Linear Discriminant Analysis (LDA) and Incremental Neighborhood Component Analysis (INCA), transform features into a new subspace that maximizes the variance between classes. However, experimental evaluations have shown that this transformation is ineffective at capturing fine-grained details and leads to information loss in complex, non-linear manifold data such as surface cracks. Furthermore, the matrix operations used in transformation and inference processes create memory requirements and computational costs.

For these reasons, a Data-Driven Selection Strategy has been adopted, which filters the most effective features without distorting their structure. This selection-based method reduces memory requirements and computational load without transforming the features. The study evaluated statistical (ANOVA F-test, f-classif) and model-based (AdaBoost) selection approaches due to the disparate structures across different domains.

Filter-Based Feature Selection (ANOVA F-test)

The ANOVA F-test (f-classif) is primarily used to rank features based on their statistical correlation with class labels. It statistically determines an attribute’s ability to distinguish between classes. Each attribute is evaluated independently (univariate) by measuring the variance between classes relative to the variance within classes. The F-score for the i -th attribute is calculated as follows:

$$F(f) = \frac{SSB/(k-1)}{SSW/(N-k)} \quad (6)$$

where SSB and SSW are the between-class and within-class sums of squares, k is the number of classes, and N is the total number of samples. Since the F-score is used solely as a ranking criterion rather than for hypothesis testing, only the relative magnitudes of the scores are considered for feature selection.

In the experiments, the k features with the highest scores according to the calculated F-scores were selected and applied to classification models. While advantageous in terms of computational cost, this method has the disadvantage of ignoring interactions between features and not accounting for non-linear decision boundaries.

Embedded Model-Based Future Selection

This model-based approach is a dynamic selection method that considers interactions between features and non-linear relationships. The AdaBoost algorithm selects features with high discriminative power/representative power when constructing decision trees during the training process. For feature selection, the Mean Importance Threshold approach calculates the arithmetic mean (μ_{imp}) of the importance scores of all features, and only features above this mean are selected. This method ensures that the model classifies based only on attributes with high information gain by selecting attributes with discriminative capability above the average.

3.2.5. Classification

The optimized feature vectors, following the hybrid feature combination and feature selection stages, are fed into classification models for surface defect detection. In this study, lightweight classifiers were preferred to avoid the computational complexity of deep neural networks and to enable inference on resource-constrained edge devices. In this context, Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Decision Tree (DT), Random Forest (RF), Logistic Regression (LR), Naive Bayes (NB), Adaptive Boosting (AdaBoost), Gradient Boosting Machine (GBM), Extra Trees (ET), and Multi-Layer Perceptron (MLP) algorithms were used to conduct experiments [83].

Furthermore, to validate the classification performance and computational efficiency of the proposed method, comparisons were made with MobileNetV2 [84], VGG16 [85], EfficientNetB0 [86], and ResNet50 [72], which are known as SOTA models in deep learning.

The performance of the proposed methodology has been evaluated primarily using standard classification metrics. Accuracy, precision, F1-score, and sensitivity were calculated based on the components of the confusion matrix: True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). Accuracy reflects the overall success of the method/system; precision indicates how many of the examples labeled as cracks by the model are actually cracks; sensitivity indicates how many of the actual cracks are detected by the model; and the F1-score is the harmonic mean of precision and sensitivity. The mathematical expressions of the metrics used are as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (7)$$

$$Precision = \frac{TP}{TP + FP} \quad (8)$$

$$Recall = \frac{TP}{TP + FN} \quad (9)$$

$$F1 - Score = 2 \times \left(\frac{Precision \times Recall}{Precision + Recall} \right) \quad (10)$$

3.3. Experimental Setup

To verify the suitability of the proposed method for resource-constrained edge devices, all experiments were performed on the CPU without using the GPU. The experiments were conducted in the Google Colab environment, using Python 3.13.15, with Scikit-learn 1.6.1, OpenCV 4.14.0, Scikit-image 0.25.2, PyWavelets 1.8.0, SciPy 1.16.3, NumPy 2.1.3, and SHAP 0.52.0. All methodological experiments on the datasets were performed under the same environmental conditions and using the same parameters.

For embedded deployment evaluation, the proposed pipeline was implemented on a Raspberry Pi 5 equipped with a quad-core ARM Cortex-A76 CPU and 8 GB of RAM, using

CPU-only execution. Inference time has been measured for a single image input (infer time = total time/repetition count) representing real-time edge device scenarios.

The inference time (t_{infer}) is taken as the end-to-end processing latency per image, encompassing the feature extraction ($t_{feature_extract}$), feature selection (masking- $t_{selection}$), and classifier prediction ($t_{predict}$) stages.

$$t_{infer} = t_{feature_extract} + t_{selection} + t_{predict} \quad (11)$$

Inference Speed (FPS) indicates the time (in milliseconds) for the trained model to produce a prediction on a single image and, consequently, the number of frames processed per second.

$$FPS = \frac{1}{t_{infer}} \quad (12)$$

4. Results and Discussion

This section presents the datasets, experimental setup, evaluation metrics, and comparative analyses used to assess the effectiveness of the proposed multi-identifier and data-driven selection framework for use on edge devices. The results are discussed in terms of classification accuracy, cross-domain robustness, feature size, and computational efficiency.

4.1. Analysis of Feature Engineering Strategies

In the first phase of the study, the effect of the Transformation and Selection-based dimensionality reduction approaches theoretically presented in Section 3.2.4 was analyzed to reduce high-dimensional feature vectors (HOG: 8100) to make them suitable for edge devices. Using images from the relatively simple egg crack dataset, the Linear Discriminant Analysis (LDA-INCA) and ANOVA-f (f-classif) methods were evaluated in terms of classification performance using machine learning models (Section 3.2.5). The comparison is presented in Table 2.

Table 2. Comparison of different feature engineering strategies on the EggCrack dataset. In this dataset, which provides high performance rates, dimension reduction methods are compared specifically for the HOG method, which extracts a large number of features. Unselected (original) classification performance and transformation-based and selection-based classification metrics are given.

Method	Dimension Reduction Strategy	Number of Features (d)	Best Model *	Test Accuracy	Test F1-Score
HOG (Original)	None-Baseline	8100	MLP/LR	0.982	0.970
HOG (AdaBoost)	Feature Selection	49	SVM	0.973	0.954
HOG (f_classif)	Feature Selection	500	SVM/AdaBoost	0.965	0.938
HOG (INCA)	Transformation	-	SVM/NB	0.754	0.559

* The models that achieved the best test results were trained with 10 different machine learning classification models.

In terms of results, the transformation-based LDA (INCA) method lost important fracture-related information while attempting to reduce the data to a linear subspace, achieving an F1-score of only 55.9%/45.8%, which is relatively low-compared to the SVM/Naive Bayes model, which performed best with an accuracy of 75.4% and 80.7%.

While the original HOG feature vector had 8100 dimensions, the statistical f-classif method achieved 96.5% success by selecting 500 features, and the model-based AdaBoost selection achieved 97.3% accuracy with 49 features. Despite this small number of features, the classification success (97.3% with SVM) is quite close to that of the original 8100-dimensional feature data (98.2%).

4.2. Comparative Analysis of Single and Hybrid Descriptors

Surface defects have different morphological characteristics depending on the material they are found in and the formation surface. The structure of cracks on a curved biological surface such as an eggshell and the crack structures on flat-textured noisy surfaces such as concrete and asphalt are characterized by sharp edges, while defects in the cell structure of industrial solar panels may manifest as textural irregularities. The performance of single descriptors (HOG, LBP, Gabor, and Wavelet) on different surface morphologies (on heterogeneous datasets) and their generalization ability were evaluated using the unselected version of the proposed hybrid method (F_{Hybrid}) on three different datasets (Table 3).

Table 3. Dataset-based performance comparison of individual descriptors and the hybrid method using 5-fold cross-validation (mean $\pm$ std).

Method	Egg Crack (Biological)	Pave Crack (Infrastructure)	ELPV (Industrial)
HOG	0.9809 $\pm$ 0.0070 (SVM)	0.8480 $\pm$ 0.0120 (SVM)	0.7691 $\pm$ 0.0058 (SVM)
LBP	0.9995 $\pm$ 0.0010 (SVM)	0.9867 $\pm$ 0.0050 (SVM)	0.7435 $\pm$ 0.0210 (SVM)
Gabor	0.9723 $\pm$ 0.0051 (MLP)	0.9147 $\pm$ 0.0115 (MLP)	0.7397 $\pm$ 0.0092 (SVM)
Wavelet	0.9940 $\pm$ 0.0020 (SVM)	0.9801 $\pm$ 0.0030 (ET)	0.7740 $\pm$ 0.0138 (RF)
Hybrid (No Selection)	0.9995 $\pm$ 0.0010 (SVM)	0.9907 $\pm$ 0.0031 (SVM)	0.7843 $\pm$ 0.0153 (SVM)

The abbreviations indicated in parentheses refer to the best classifier. The hybrid method refers to a fusion strategy at the descriptor level that combines features extracted using LBP, Gabor filters, and DWT.

The HOG descriptor, which performs well on edge gradients, demonstrated successful performance in cases in which the crack shape was more prominent (Egg Crack, 98.09%); however, it yielded moderate results in ELPV, where crystalline structures predominate. In the ELPV dataset, HOG achieved an accuracy rate of 76.91%, outperforming the LBP and Gabor methods, though slightly behind the Wavelet method. However, in the Pave Crack dataset, in which the background texture is complex and noisy, its accuracy drops to 84.80%, much lower than that of other single classifiers such as LBP and Wavelet. This indicates that HOG struggles to handle textural noise and does not possess the same sensitivity as LBP and Wavelet. Although LBP is successful at capturing local texture changes, Gabor at capturing orientations, and Wavelet at capturing frequency components, none of them alone can provide the stability demonstrated by the hybrid method, which consistently achieves the highest accuracy across all datasets.

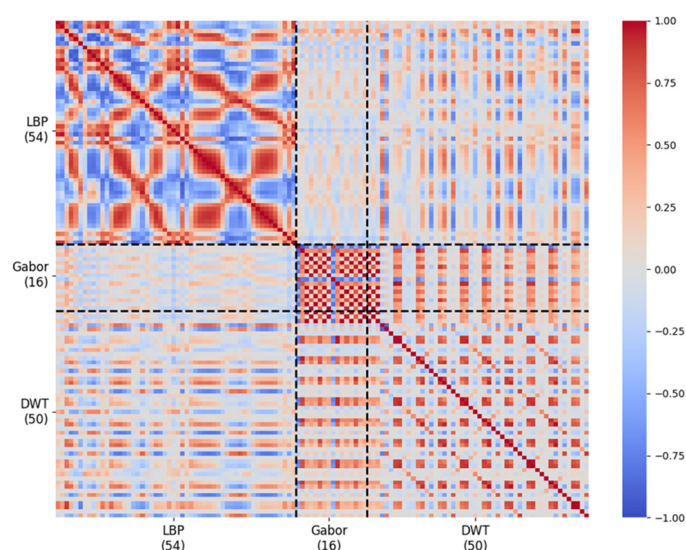
The proposed hybrid method compensates for detection success by playing a complementary role in situations where individual methods are weak. The hybrid method efficiently combines the discriminative power of LBP in surface microstructure with the success of Gabor in capturing orientation and Wavelet in capturing frequency details. It achieves higher or the same accuracy as the individual methods in all three datasets. These findings demonstrate that the method has a positive effect on crack detection by combining different feature spaces and that it has generalization capabilities.

To investigate the complementarity of feature sets in more detail, pairwise combinations of LBP, Gabor, and DWT were evaluated on the ELPV dataset using five-fold cross-validation with ten different classifiers (Table 4).

Table 4. ELPV dataset-based 5-fold cross-validation performance of different classifiers using pairwise combinations of LBP, Gabor, and DWT descriptors (mean $\pm$ std).

Classifier	Metrics	LBP + Gabor	LBP + DWT	Gabor + DWT
Random Forest	Accuracy	76.22 $\pm$ 1.50	77.06 $\pm$ 1.25	76.75 $\pm$ 1.11
	F1-score	75.20 $\pm$ 1.60	76.12 $\pm$ 1.31	75.88 $\pm$ 1.05
Extra Trees	Accuracy	76.56 $\pm$ 1.21	77.06 $\pm$ 1.80	77.02 $\pm$ 0.71
	F1-score	75.57 $\pm$ 1.33	76.06 $\pm$ 1.90	76.14 $\pm$ 0.77
MLP	Accuracy	75.57 $\pm$ 1.53	76.26 $\pm$ 1.13	73.74 $\pm$ 0.91
	F1-score	74.60 $\pm$ 1.75	75.56 $\pm$ 1.20	73.08 $\pm$ 0.93
SVM	Accuracy	75.04 $\pm$ 1.89	75.95 $\pm$ 1.86	75.46 $\pm$ 1.27
	F1-score	73.56 $\pm$ 2.06	74.78 $\pm$ 1.93	74.30 $\pm$ 1.31
KNN	Accuracy	75.95 $\pm$ 1.47	75.12 $\pm$ 1.33	74.39 $\pm$ 1.05
	F1-score	74.82 $\pm$ 1.59	73.69 $\pm$ 1.43	73.11 $\pm$ 1.15

Tree-based ensemble methods, particularly Extra Trees and Random Forest, consistently outperformed other classifiers across all feature combinations. The highest classification accuracy (77.06%) was achieved with the LBP + DWT fusion using Extra Trees. Similarly, the Gabor + DWT combination also achieved an accuracy of 77.02% with Extra Trees, demonstrating the effectiveness of combining complementary texture and frequency-domain information for surface defect classification. The performance of the LBP + DWT pair indicates that the local texture information captured by LBP and the spatial frequency information provided by DWT complement each other effectively. This observation is further supported by the correlation heatmap in Figure 3, where the cross-group regions generally exhibit weaker correlations than the within-group regions, suggesting limited redundancy and partially complementary information among the feature representations. More pronounced correlation patterns are observed between some Gabor and DWT features, indicating a certain degree of overlap between these two representations. This partial redundancy may have contributed to the marginally lower performance of the Gabor + DWT combination compared with LBP + DWT.

**Figure 3.** Heatmap of the mean absolute correlations among the LBP, Gabor, and DWT feature sets. The dotted lines indicate the boundaries between the three descriptor groups.

Furthermore, Figure 4 illustrates the structural efficiency of the proposed hybrid feature engineering method compared to traditional methods. The standard HOG descriptor generates an 8100-dimensional feature vector per 128×128 -pixel image. The hybrid feature pool, carefully constructed from components obtained based on specific parameters of LBP, Gabor, and Wavelet descriptors, is represented by a compact 120-dimensional feature vector. Thus, the proposed hybrid descriptor achieves a 98.5% reduction in feature dimensionality, corresponding to a 67.5-fold smaller representation than HOG, while maintaining competitive classification performance. It also achieved statistically comparable results on challenging image structures, such as the crystalline textures present in photovoltaic cells. These findings demonstrate the potential of the proposed representation for resource-constrained edge devices with limited memory and computational capabilities. Nevertheless, further reducing the feature dimensionality is desirable to minimize computational and memory overhead, particularly for battery-powered edge devices operating under stringent latency and energy constraints.

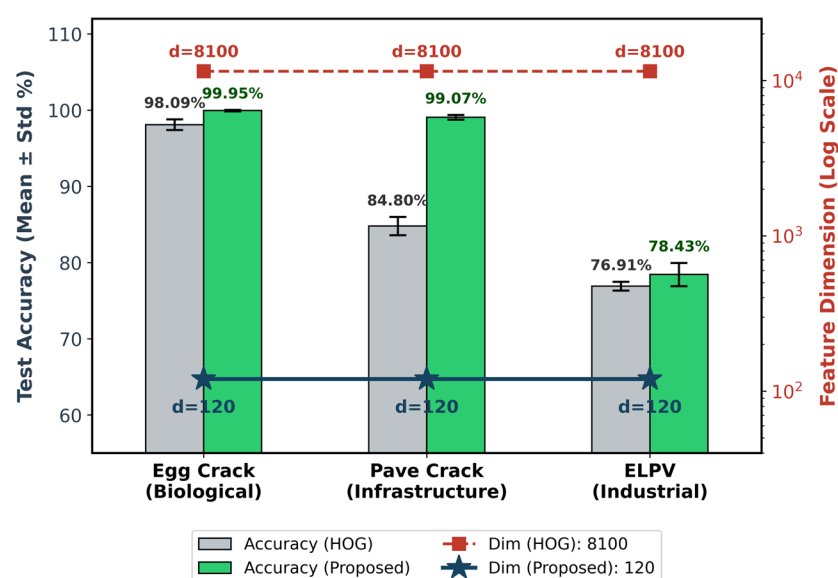


Figure 4. Accuracy and feature dimension analysis between the proposed hybrid method and the HOG descriptor. The red and blue lines represent the feature size on a logarithmic scale for each dataset. Particularly in the Pavement Crack dataset, the feature vector size extracted using the Hybrid method corresponds to approximately 98% less than the HOG feature vector size and achieves high accuracy gains.

4.3. Impact of Data-Driven Selection

Feature dimension and computational cost are important parameters for resource-constrained devices. These parameters affect the deployment of the model to edge devices, as well as the subsequent model training and defect inference stages from images. Statistical and model-based selection methods were applied to the proposed hybrid feature pool to improve compatibility with resource-constrained edge devices without compromising performance. Table 5 presents a comparative overview of the best classification scores for the unselected full hybrid feature pool (no-selection) and the applied selection strategies for each of the three datasets.

Table 5. Dataset-based performance comparison of different selection strategies on hybrid feature pool.

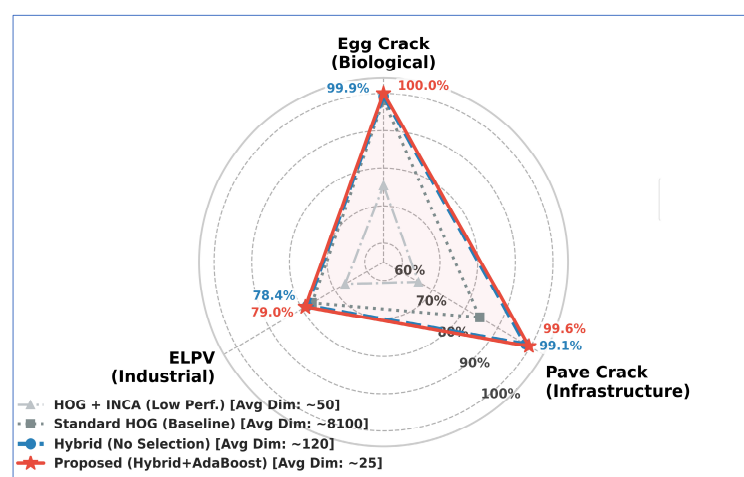
Dataset	Classification Model *	No Selection ($d = 120$) *	AdaBoost Selection ($d = 25$) *	f-Classif Selection ($k = 50$) *
Egg Crack	SVM	0.9995 ± 0.0010 (SVM)	1.00 ± 0.00 (SVM)	1.00 ± 0.00 (SVM)
Pave Crack	SVM	0.9907 ± 0.0031 (SVM)	0.9960 ± 0.0021 (SVM)	0.9891 ± 0.0064 (SVM)
ELPV	SVM	0.7843 ± 0.0153 (SVM)	0.79 ± 0.0108 (SVM)	0.7668 ± 0.0157 (SVM)

* The results were reported using the DT, RF, ET, and SVM classifiers, which showed the best performance among the evaluated classification models. Feature dimensions are denoted by d and k .

Experimental results show that the AdaBoost-based embedded selection strategy outperforms other methods regardless of surface defect types. When hybrid feature vectors ($d = 120$) were used on the ELPV dataset, the accuracy value of 78.43% decreased to 76.68% with the f-classif ($k = 50$) selection method. The model-based AdaBoost selection method selected 25 effective features, increasing the accuracy to 79.0%. This relative increase indicates that the AdaBoost selection method successfully identified features that capture the distinction between crystalline texture and cracks while eliminating features with very small effective features.

Looking at the results obtained on the EggCrack and PaveCrack datasets, it is observed that the model-based AdaBoost selection method is more successful than the hybrid feature set and the f-classif selection method. Although the proposed AdaBoost selection method reduced the feature vector dimension by 80%, it still achieved an increase in accuracy across all datasets. This result confirms that the proposed method can achieve its goal of implementation to resource-constrained edge devices.

To further illustrate the relationship between feature dimension reduction and classification performance more clearly, Figure 5 presents the comprehensive experimental results obtained with three different datasets containing selection methods as a radar chart. In this illustration, the radial axes show the test accuracy for each dataset, while the legend indicates the average feature size for each method.

**Figure 5.** Efficiency-Performance analysis between proposed hybrid-based methods and HOG-based methods.

The proposed Hybrid feature and AdaBoost selection method (red solid line) covers the largest triangular area in the graph and demonstrates more successful and consistent performance across all domains (Smooth, Textured, and Complex). Particularly in the Pave Crack dataset, while HOG-based methods show central narrowing, the proposed method achieves high accuracy ($>96\%$) across all axes. However, it was observed that the

INCA transformation applied to the HOG feature space significantly reduced accuracy. Selection approaches according to the HOG-based INCA transformation maintained or improved accuracy. While HOG-based methods showed fluctuations across different datasets, hybrid methods exhibited stable performance within a more limited framework. The proposed selection strategy ($d = 25$) successfully stabilized the model, demonstrating edge compatibility in a balance between high accuracy (wide coverage) and low dimensionality.

To analyze the noise robustness and classification performance of the proposed hybrid selection method, we examined the confusion matrices on the ELPV dataset (Figure 6), which presents higher detection difficulty due to its noisy crystalline background. When comparing the HOG-based model ($d = 8100$) with our AdaBoost-selected hybrid model ($d = 25$), the HOG approach achieves a slightly lower false-positive rate (185 vs. 273), indicating a more conservative prediction behavior. However, this conservatism comes at the cost of missing a significantly larger number of actual defects (FN: 407 vs. 278). In contrast, the proposed method correctly identifies 129 additional real defects, achieving a much higher recall (75.1% vs. 63.5%) and superior overall accuracy (79.00% vs. 77.44%)—all while using a fraction of the feature size (25 vs. 8100). This trade-off is particularly desirable in safety-critical edge applications, where missing a defect is far more costly than a false alarm.

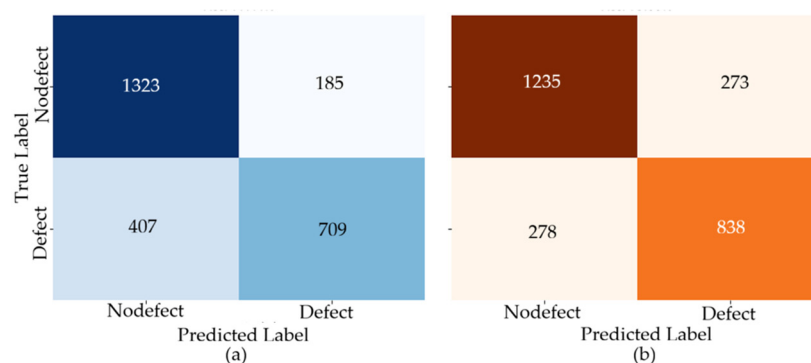


Figure 6. Confusion matrix analysis with the ELPV dataset: (a) high-dimensional HOG descriptor ($d = 8100$) and (b) proposed AdaBoost selective hybrid method ($d = 25$).

The ROC analysis (Figure 7) further supports these findings. The proposed method achieves an AUC of 0.8546, slightly higher than the HOG-based SVM baseline (AUC = 0.8499). While the improvement in AUC is modest, it is important to emphasize that this performance is attained with a 324-fold reduction in feature dimensionality (25 vs. 8100).

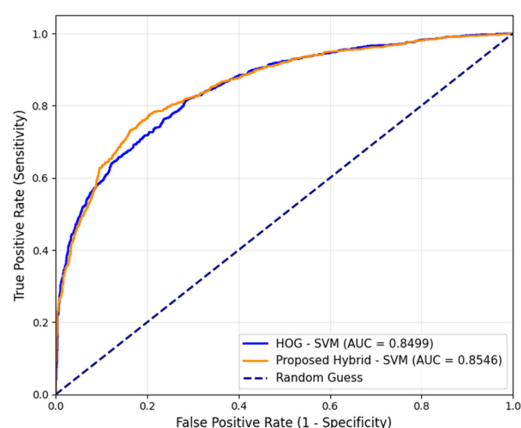


Figure 7. ROC-AUC performance analysis: Comparison between the baseline high-dimensional feature set and the proposed AdaBoost-based selective hybrid method.

The ROC curves also confirm that the proposed method maintains a higher true positive rate across a wide range of false positive thresholds, indicating better separability between defect and non-defect patterns. This is particularly evident in the ELPV dataset, where the proposed model effectively suppresses false alarms caused by the noisy crystalline structure of solar cells, achieving a more reliable distinction of non-defective regions. The slight AUC gain, combined with the drastic reduction in feature size, further validates the proposed method as a highly efficient and robust solution for edge deployment.

Stability and Generalization of the Selected Features

To further validate the robustness of the AdaBoost-based feature selection, we conducted two additional analyses on the ELPV dataset.

First, we performed a bootstrap stability analysis using the Jaccard similarity index. The AdaBoost selection process was repeated 10 times with different random seeds, and the overlap among the selected 25-feature subsets was measured. The mean Jaccard index was 0.906 ± 0.041 , indicating that the selected features are highly consistent (over 90% overlap) across independent runs. This confirms that the AdaBoost importance scores are stable and suitable as a global selection criterion.

Second, we applied a nested 5×3 cross-validation scheme to completely eliminate any data leakage between feature selection and final evaluation. In the outer loop (5 folds), a held-out test set was reserved; in the inner loop (3 folds), AdaBoost selection and hyperparameter tuning (SVM: C, gamma) were performed on the remaining training data. The nested CV yielded an accuracy of $77.55\% \pm 1.47\%$ and an F1-score of $77.55\% \pm 1.46\%$ on ELPV. These results are nearly identical to the 5-fold CV accuracy we previously reported. This confirms that our findings were not affected by data leakage and are fully reproducible. The results are presented in Table 6. Taken together, these analyses validate the robustness and reliability of the feature selection and evaluation process.

Table 6. Stability analysis and nested cross-validation results for the AdaBoost-selected 25-feature subset on the ELPV dataset.

Analysis	Metric	Result
Bootstrap Stability (Jaccard Index)	Mean $\pm$ std over 10 runs	0.906 ± 0.041
Nested 5×3 CV	Accuracy (%)	77.55 ± 1.47
Nested 5×3 CV	F1-Score (%)	77.55 ± 1.46

Jaccard index > 0.8 indicates stable feature selection. Nested CV results confirm that no information leakage occurred.

4.4. Effect of Input Resolution on Classification Performance

To evaluate the impact of input resolution on the proposed framework, we performed an ablation study on the ELPV dataset using three different image sizes: 64×64 , 128×128 , and 224×224 . The experiments were conducted with the AdaBoost-selected feature subset, followed by classification with ML models under 5-fold cross-validation. As shown in Table 7, the highest accuracy ($77.44\% \pm 1.00\%$) was achieved at 128×128 resolution with the AdaBoost + SVM configuration. Interestingly, increasing the resolution to 224×224 did not improve accuracy; instead, it resulted in a slight drop to $75.53\% \pm 0.84\%$, while also increasing inference time. At 64×64 , the accuracy was comparable ($77.21\% \pm 1.40\%$) but with slightly higher variance, indicating fewer stable predictions.

Table 7. Ablation study of input resolution for the proposed AdaBoost-selected hybrid framework on the ELPV dataset.

Input Size	Selection Method	Classifier	Accuracy (%) *	F1-Score (%) *	Inference Time (ms/image) *
64 × 64	AdaBoost	ExtraTrees	77.21 ± 1.40	77.00 ± 1.44	0.2068
64 × 64	AdaBoost	SVM	77.25 ± 1.53	77.22 ± 1.51	0.1189
128 × 128	AdaBoost	SVM	79.00 ± 1.08	78.99 ± 1.09	0.1034
128 × 128	AdaBoost	MLP	77.82 ± 1.23	77.70 ± 1.25	0.2202
224 × 224	AdaBoost	SVM	75.95 ± 1.20	75.92 ± 1.18	0.1163
224 × 224	AdaBoost	ExtraTrees	75.53 ± 0.84	75.22 ± 0.84	0.2294

* Results are reported as mean ± std over 5-fold cross-validation.

These results confirm that 128×128 provides the optimal trade-off between classification accuracy, prediction stability, and computational efficiency. The marginal gain from higher resolution is outweighed by the increased computational cost, and the lower resolution (64×64) sacrifices stability without significant speed benefits. Therefore, we adopted 128×128 as the default input size for all experiments, ensuring both high performance and real-time suitability for resource-constrained edge devices.

4.5. Comparison with Deep Learning Models

Deep learning models have demonstrated highly successful performance in detecting surface defects in recent years. To achieve this success, millions of parameters are used and intensive computations are performed. The proposed lightweight hybrid framework has been compared with widely known deep learning models in terms of accuracy, model size, and computational costs.

The ELPV dataset, which is a challenging scenario for the study, was used for comparison. Deep learning models were pre-trained with ImageNet weights and fine-tuned for the dataset. The models were trained for 40 epochs using the Adam optimizer ($\text{lr} = 1 \times 10^{-3}$) and cross-entropy loss. Model checkpoints were saved at the epoch with the highest validation accuracy. The data was split using 5-fold stratified cross-validation, with a batch size of 64 and num_workers set to 2. In addition, the number of parameters and FLOPs were calculated to quantitatively assess the computational complexity of the models. FLOPs were estimated by multiplying the number of multiply–accumulate operations (MACs), obtained using torchprofile, by two ($\text{FLOPs} = 2 \times \text{MACs}$). The proposed method employed an RBF-SVM classifier, with hyperparameters optimized via grid search over $C = \{1, 10, 50, 100\}$ and $\gamma = \{\text{scale}, 0.01, 0.001\}$ using 3-fold cross-validation. Stratified 5-fold partitioning, class_weight = ‘balanced’, random_state = 42, and parallel processing (n_jobs = −1) were employed to ensure reproducible and computationally efficient model optimization. Comparative quantitative results are presented in Table 8, and the complexity-efficiency visualization is presented in Figure 8.

As shown in Table 8, the ResNet50 model has the highest accuracy, but it requires approximately 90 MB of memory storage and incurs high computational costs, requiring millions of floating-point operations. This makes the model unsuitable for use on microcontrollers such as ESP32 and STM32. Thanks to its compact structure, the proposed hybrid framework achieves accuracy ($\pm 1\text{--}2.3\%$) that can almost compete with complex deep learning models, with a very small model size and computational cost. It demonstrates higher accuracy with smaller model size compared to lightweight DL models such as MCUNet-In0, MobileNetV2 and EfficientNetV2B0, which are specifically designed for mobile platforms.

Table 8. Comparison of the proposed method and deep learning models in the ELPV dataset at 128×128 resolution.

Model	Architecture/Backbone	Accuracy (%) *	F1-Score (%) *	Parameters(M)	Model Size (MB)	FLOPs * (M)
EfficientNetV2-B0	Lightweight CNN	83.16 ± 1.52	82.34 ± 1.86	5.75	22.90	424.31
MNASNet-100	Lightweight CNN	82.85 ± 1.00	82.13 ± 1.04	3.07	12.20	204.55
MobileNetV2	Lightweight CNN	81.52 ± 1.24	80.76 ± 1.37	2.19	8.84	195.64
MCUNet-In0	Lightweight CNN	78.16 ± 2.47	77.33 ± 2.65	0.56	2.30	89.51
ResNet50	Heavy CNN	83.19 ± 0.33	82.51 ± 0.40	23.51	90.03	2698.26
VGG16	Heavy CNN	80.41 ± 3.10	79.12 ± 3.65	134.28	512.29	10296.79
ConvNeXtTiny	Heavy CNN	71.52 ± 2.19	69.51 ± 2.92	27.80	106.28	2909.26
Proposed Method	Optimized hybrid + SVM	79.00 ± 1.08	78.99 ± 1.09	0.029	0.256	0.056

* Results are reported as mean $\pm$ std over 5-fold cross-validation with an input resolution of 128×128 . FLOPs values are calculated for the inference phase. In CNN models, FLOPs are calculated based on the total number of multiply-accumulate operations for the entire network architecture for a fixed input size; for SVM, FLOPs are calculated using the formula $2 \times n_{\text{support_vectors}} \times n_{\text{features}}$, and the exponential operations of the kernel function are not included in this metric.

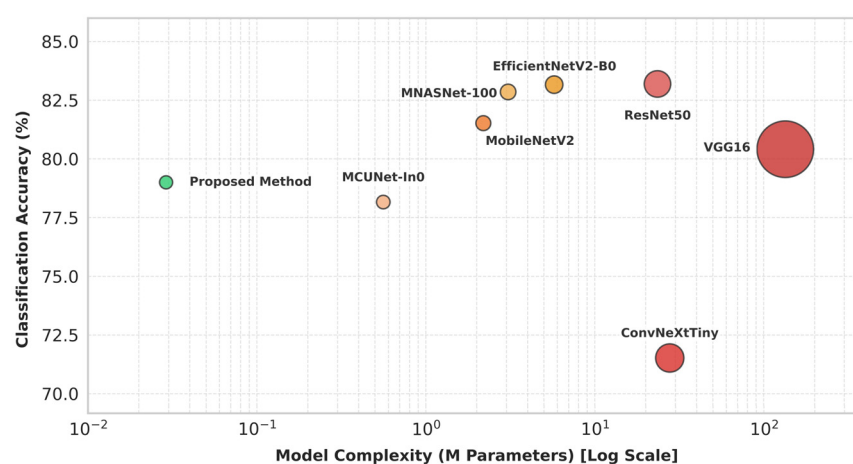


Figure 8. Complexity vs. Accuracy analysis of deep learning models and the proposed hybrid framework on the ELPV dataset. The x-axis shows the number of parameters (in millions) on a logarithmic scale. The proposed method (green circle) outperforms Lightweight CNN models with near-zero parameter cost.

The logarithmic scale in Figure 8 highlights the efficiency difference between the proposed framework and deep learning models. When compared to the MCUNet-In0 model—a highly compact CNN adapted for microcontrollers—the proposed framework demonstrates significant performance in efficiency metrics (approximately 19 times fewer parameters at 0.029 M; 9 times smaller model size at 0.256 MB). In terms of computational cost, it requires over 1500 times less computational cost, at 0.056M FLOPs. In terms of detection performance, the proposed method also achieves higher accuracy (78.16% vs. 79.00%) and F1-score (77.33% vs. 78.99%) compared to MCUNet-In0. However, while the ResNet50 model—which demonstrates the highest accuracy—remains at a high complexity level on the right side of the graph, the proposed framework is positioned in the upper-left corner with nearly zero complexity/cost. At this point, the proposed model demonstrates a good balance between computational efficiency and detection accuracy; particularly in real-time industrial applications using edge devices, it offers significant potential in terms of deployability, energy consumption, and computational efficiency without compromising accuracy to a great extent.

4.6. Computational Efficiency and Edge Device Analysis

To evaluate the real-world applicability of the proposed method for edge devices, inference latency was benchmarked against state-of-the-art deep learning models on a Raspberry Pi 5 (CPU-only). Since the computational complexity of feature extraction algorithms depends on image resolution rather than texture content, measurements were performed on the ELPV dataset (resized to 128×128 pixels) to represent all domains. As detailed in Table 9, deep learning models typically operate in an end-to-end manner, where the heavy matrix multiplications (convolutions) consume the majority of the processing time, ranging from 49.401 ms (MCUNet-In0) to 583.709 ms (VGG16).

Table 9. Inference latency and frame rate comparison on CPU-only (Raspberry Pi 5) for 128×128 input resolution.

Method/Model	Feature Extraction (ms) *	Classification (ms) *	Total Inference Time (ms) *	FPS
VGG16	—	—	583.709	~1.7
ConvNeXtTiny	—	—	174.687	~5.7
ResNet50	—	—	158.840	~6.2
EfficientNetV2B0	—	—	84.762	~11.7
MobileNetV2	—	—	77.945	~12.8
MNASNet-100	—	—	59.237	~16.8
MCUNet-In0	—	—	49.401	~20.2
Proposed (Hybrid feature+ AdaBoost + SVM)	36.692	0.365	37.057	~26.9

* All results are the arithmetic mean of 100 measurements taken after 20 warm-up steps. The feature extraction time includes the total duration of image resizing, LBP, Gabor, and DWT feature extraction, feature selection, and data scaling (32.779 ms + 3.187 ms + 0.324 ms + 0.402 ms).

The proposed (AdaBoost $d = 25$) method has an end-to-end detection time of approximately 37 ms. Of this time, 32.77 ms is spent on hybrid feature extraction, 3.187 ms on feature vector selection, and 0.365 ms on SVM classifier estimation. The end-to-end detection time corresponds to an inference rate of approximately 27 FPS, meeting the standard requirement for real-time detection systems.

These findings align with the fact that by replacing heavy deep neural network layers with carefully selected, hand-designed features and efficient classifiers, real-time detection speeds can be achieved on standard microcontroller-class CPUs without the need for specialized accelerators such as GPUs or TPUs.

Training the SVM model with features selected from the hybrid feature pool obtained from images in the ELPV dataset training section took 1.69 s (Google Colab, CPU-only). Considering the training cost of deep learning models, which can take hours, this is a very small value and constitutes an advantage. This low training cost has the potential for continuous learning and, in particular, federated learning applications distributed on IoT devices.

4.7. Explainability and Computational Feature Efficiency

The SHAP analysis presented in Figure 9a,b provides feature-level transparency into the decision-making process of the proposed framework. As shown by the global feature importance ranking, influential features originate from all three descriptor families, including fine-grained local texture patterns captured by LBP (e.g., LBP_P8_R1), orientation-sensitive responses characterized by Gabor features (e.g., Gabor_T135_Std), and statistical properties of multiscale frequency components derived from DWT (e.g., DWT_ch3_Kurtosis). This distribution indicates that the classifier does not rely exclusively on a single descriptor

domain but instead exploits information from complementary texture, orientation, and frequency representations.

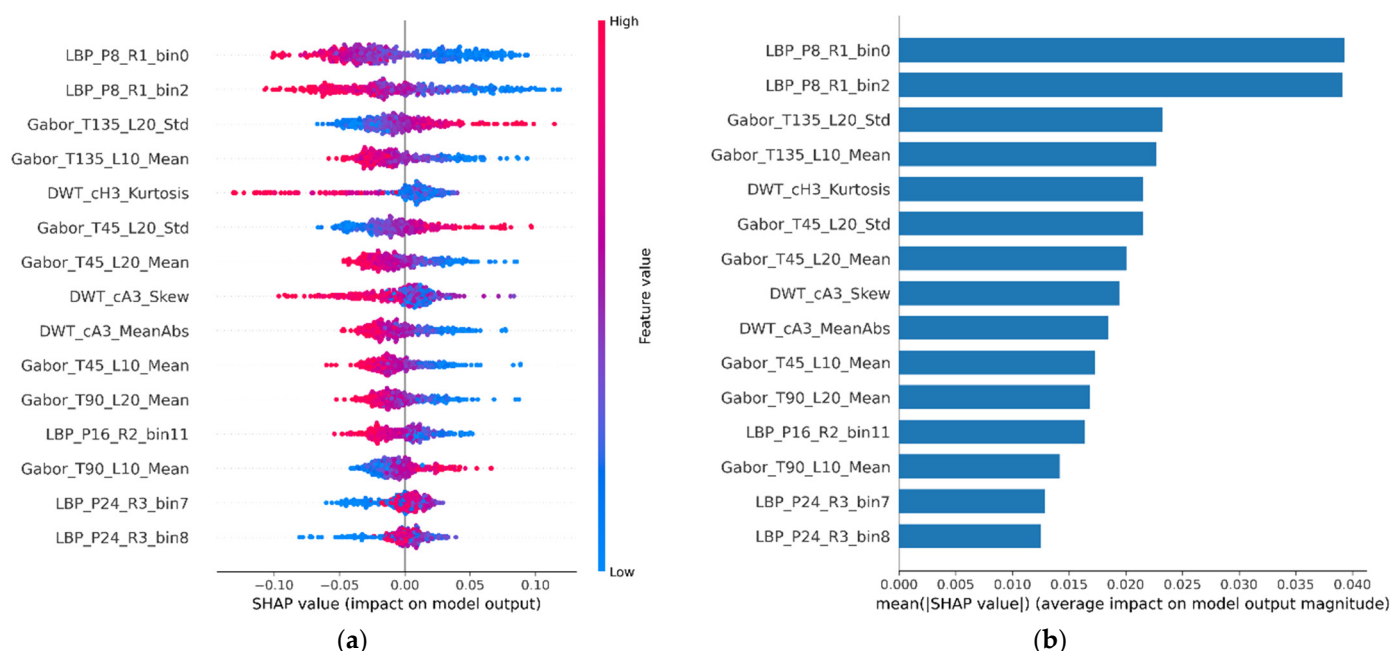


Figure 9. SHAP-based feature attribution analysis of the proposed data-driven framework: (a) Bee swarm plot of SHAP value distributions. (b) Global feature importance ranking.

Given that the input features are global statistical descriptors rather than pixel-level activations, SHAP analysis provides a direct feature-based evaluation rather than spatial localization. Thus, SHAP values explain the contribution of handcrafted features to the output rather than providing spatial maps of defective regions.

From an edge-computing perspective, the SHAP results reveal that discriminative information can be represented via computationally efficient handcrafted features: LBP captures local microtexture, Gabor provides orientation/frequency selectivity, and DWT supplies multiscale frequency content. Unlike CNNs, which learn analogous representations through costly convolutional operations, the proposed framework exploits these features without intensive computation. Alongside classification and complexity metrics, SHAP confirms that the framework leverages informative low-level texture, orientation, and frequency characteristics without relying on deep feature learning. Moreover, AdaBoost-based selection reduced the feature set from 120 to 25 (79.2% reduction), retaining only the most discriminative features. This compact, interpretable strategy minimizes computational load and memory footprint, making it ideal for resource-constrained edge environments.

4.8. Limitations of the Study

The proposed method offers a resource-efficient and sustainable alternative for edge deployment, particularly in scenarios where deep learning-based solutions are impractical due to hardware and energy constraints.

While the proposed framework demonstrates significant advantages in terms of explainability and computational efficiency on resource-constrained edge devices, it has some limitations. The robustness of the system under outdoor-specific, variable, and uncontrolled lighting conditions has not been fully verified compared to DL models with higher learning capabilities.

Runtime performance and latency measurements were conducted on the Raspberry Pi 5 CPU edge device and in the Python environment. Although the requirements are

theoretically met for microcontrollers, the framework has not yet been practically deployed and tested on ultra-low-power microcontrollers (e.g., ESP32) or bare-metal hardware environments like FPGAs. Considering resource-constrained devices, the input image size was set to 128×128 pixels in the study. While this resolution is sufficient for the evaluated datasets, detecting micro-scale surface defects in more complex industrial scenarios may require higher-resolution inputs (e.g., 640×640). Because the computational cost of the proposed hybrid descriptors scales directly with pixel density, processing larger input sizes could introduce new latency bottlenecks, which remains an area for future optimization.

Additionally, dataset-specific limitations should be noted. For EggCrack, the non-defective and defective images originate from different sources, which may introduce a domain shift—though our preprocessing and texture-based features reduce its impact. Class imbalance exists in the EggCrack dataset. We did not apply oversampling or class weighting, as stratified CV and F1-score already provide a reliable evaluation under the natural distribution, and additional balancing would contradict the efficiency goals of edge deployment. For PaveCrack, original resolutions differ by class (227×177 vs. 640×640), but all images were uniformly resized to 128×128 , eliminating resolution-based bias. For ELPV, the binary threshold (0.33, 0.66, 1.0 vs. 0.0) follows the original multi-level labels and common literature practice, yet no manual expert verification was conducted.

5. Conclusions

This study proposes an intelligent, explainable, and data-driven hybrid feature extraction framework that addresses the trade-off between predictive performance and real-time computational efficiency for resource-constrained edge devices. To overcome the inherent “black-box” nature and high computational demands of modern deep learning models, as well as the limited representational capacity of traditional single descriptors, we introduced a complementary hybrid feature extraction architecture using LBP (Pattern), Gabor (Direction), and Wavelet (Frequency) descriptors and subsequently optimized through a model-based (AdaBoost) feature selection strategy.

On the challenging ELPV dataset, the proposed method achieved a classification accuracy of $79.00 \pm 1.08\%$ and an F1-score of $78.99 \pm 1.09\%$. Although its accuracy was lower than that of several CNN baselines, it outperformed MCUNet-In0 and ConvNeXtTiny while requiring substantially fewer computational resources. Specifically, the proposed method used only 0.029 M parameters, occupied 0.256 MB of model storage, and required 0.056 M FLOPs. With an end-to-end inference time of 37 ms (~ 26 FPS) on an ARM Cortex-A76 CPU and minimal RAM usage, the system meets the real-time requirements of industrial production without requiring expensive hardware accelerators or external memory. Furthermore, SHAP-based analysis improved feature-level transparency by quantifying the contributions of individual handcrafted descriptors and statistical features to the classification decisions, enabling the predictions to be interpreted in terms of low-level texture, orientation, and frequency characteristics rather than relying solely on opaque neural representations.

Future work plans include optimizing the algorithm using C++ and quantization techniques for deployment on embedded microcontrollers such as ESP32 and FPGAs and testing it in autonomous structural inspection scenarios by integrating it into unmanned vehicles. Additionally, we aim to establish federated learning by leveraging the computational efficiency and low communication cost advantages of the proposed approach.

Funding: This research received no external funding.

Data Availability Statement: The datasets analyzed in this study are publicly available from Roboflow Universe and Kaggle. Access details are provided in the References section.

Acknowledgments: During the preparation of this manuscript, the author used Google Gemini (Gemini 3.1 Pro) for the purposes of sentence structure editing. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The author declares no conflicts of interest.

References

1. Mittal, P. A Comprehensive Survey of Deep Learning-Based Lightweight Object Detection Models for Edge Devices. *Artif. Intell. Rev.* **2024**, *57*, 242. [[CrossRef](#)]
2. Saha, S.; Xu, L. Vision Transformers on the Edge: A Comprehensive Survey of Model Compression and Acceleration Strategies. *Neurocomputing* **2025**, *643*, 130417. [[CrossRef](#)]
3. Tabanelli, E.; Brunelli, D.; Acquaviva, A.; Benini, L. Trimming Feature Extraction and Inference for MCU-Based Edge NILM: A Systematic Approach. *IEEE Trans. Ind. Inform.* **2021**, *18*, 943–952. [[CrossRef](#)]
4. Qiao, Q.; Ahmad, A.; Hu, H.; Wang, K. Advancements in Industrial Product Surface Defect Detection: From Traditional Methods to Modern Advanced Techniques. In *Machine Learning and Intelligent Systems: Proceedings of MLIS 2024*; IOS Press: Amsterdam, The Netherlands, 2024; pp. 82–89.
5. Cumbajin, E.; Rodrigues, N.; Costa, P.; Miragaia, R.; Frazão, L.; Costa, N.; Fernández-Caballero, A.; Carneiro, J.; Buruberri, L.H.; Pereira, A. A Systematic Review on Deep Learning with CNNs Applied to Surface Defect Detection. *J. Imaging* **2023**, *9*, 193. [[CrossRef](#)] [[PubMed](#)]
6. Atasoy, H.; Kutlu, Y. CNNFET: Convolutional Neural Network Feature Extraction Tools. *SoftwareX* **2025**, *30*, 102088. [[CrossRef](#)]
7. Paradisa, R.H.; Sarwinda, D.; Bustamam, A.; Argyadiva, T. Classification of Diabetic Retinopathy through Deep Feature Extraction and Classic Machine Learning Approach. In *Proceedings of the 2020 3rd International Conference on Information and Communications Technology (ICOIAC)*; IEEE: Piscataway, NJ, USA, 2020; pp. 377–381.
8. Suyahman, S.; Hapsari, A. VGG-Based Feature Extraction for Classifying Traditional Batik Motifs Using Machine Learning Models. *Preserv. Digit. Technol. Cult.* **2025**, *54*, 215–222. [[CrossRef](#)]
9. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale. *arXiv* **2020**, arXiv:2010.11929.
10. Touvron, H.; Cord, M.; Douze, M.; Massa, F.; Sablayrolles, A.; Jégou, H. Training Data-Efficient Image Transformers & Distillation through Attention. In *Proceedings of the International Conference on Machine Learning (ICML)*; PMLR: New York, NY, USA, 2021; pp. 10347–10357.
11. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*; IEEE: Piscataway, NJ, USA, 2021; pp. 10012–10022.
12. Dhal, P.; Azad, C. A Comprehensive Survey on Feature Selection in the Various Fields of Machine Learning. *Appl. Intell.* **2022**, *52*, 4543–4581. [[CrossRef](#)]
13. Deng, L.; Li, H.; Liu, H.; Zhang, H.; Zhao, Y. Research on Multi-Feature Fusion for Support Vector Machine Image Classification Algorithm. In *Proceedings of the 2021 IEEE International Conference on Electronic Technology, Communication and Information (ICETCI)*; IEEE: Piscataway, NJ, USA, 2021; pp. 516–519.
14. Liao, J.; Guo, L.; Jiang, L.; Yu, C.; Liang, W.; Li, K.; Pop, F. A Machine Learning-Based Feature Extraction Method for Image Classification Using ResNet Architecture. *Digit. Signal Process.* **2025**, *160*, 105036. [[CrossRef](#)]
15. Markovic, T.; Lidholm, P.; Strandberg, P.E.; Leon, M. Feature Selection Using Genetic Algorithm for Intrusion Detection on Resource-Constrained Edge Devices. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*; ACM: New York, NY, USA, 2025; pp. 1415–1423.
16. Fatima, M.; Rehman, O.; Rahman, I.M.; Ajmal, A.; Park, S.J. Towards Ensemble Feature Selection for Lightweight Intrusion Detection in Resource-Constrained IoT Devices. *Future Internet* **2024**, *16*, 368. [[CrossRef](#)]
17. Tsai, T.H.; Tung, N.C.; Chen, C.Y. An FPGA-Based Reconfigurable Convolutional Neural Network Accelerator for Tiny YOLO-V3. *Circ. Syst. Signal Process.* **2025**, *44*, 3388–3409. [[CrossRef](#)]
18. Khataei, A.; Bazargan, K. TreeLUT: An Efficient Alternative to Deep Neural Networks for Inference Acceleration Using Gradient Boosted Decision Trees. In *Proceedings of the 2025 ACM/SIGDA International Symposium on Field Programmable Gate Arrays (FPGA)*; ACM: New York, NY, USA, 2025; pp. 14–24.
19. Trigka, M.; Dritsas, E. A Comprehensive Survey of Deep Learning Approaches in Image Processing. *Sensors* **2025**, *25*, 531. [[CrossRef](#)] [[PubMed](#)]
20. Kang, Z.; Zhao, Y.; Chen, L.; Guo, Y.; Mu, Q.; Wang, S. Advances in Machine Learning and Hyperspectral Imaging in the Food Supply Chain. *Food Eng. Rev.* **2022**, *14*, 596–616. [[CrossRef](#)] [[PubMed](#)]

21. Nahar, L.; Awrangjeb, M.; Islam, M.S. AI-Enabled Defect Detection in Industrial Products: A Comprehensive Survey, Key Insights and Future Research Challenges. *Adv. Eng. Inform.* **2026**, *69*, 104067. [[CrossRef](#)]
22. Tang, B.; Chen, L.; Sun, W.; Lin, Z.K. Review of Surface Defect Detection of Steel Products Based on Machine Vision. *IET Image Process.* **2023**, *17*, 303–322. [[CrossRef](#)]
23. Azouz, Z.; Honarvar Shakibaei Asli, B.; Khan, M. Evolution of Crack Analysis in Structures Using Image Processing Technique: A Review. *Electronics* **2023**, *12*, 3862. [[CrossRef](#)]
24. Safaei, N.; Smadi, O.; Masoud, A.; Safaei, B. An Automatic Image Processing Algorithm Based on Crack Pixel Density for Pavement Crack Detection and Classification. *Int. J. Pavement Res. Technol.* **2022**, *15*, 159–172. [[CrossRef](#)]
25. Zheng, Y.; Li, S.; Xiang, Y.; Zhu, Z. Crack Defect Detection Processing Algorithm and Method of MEMS Devices Based on Image Processing Technology. *IEEE Access* **2023**, *11*, 126323–126334. [[CrossRef](#)]
26. Zhou, X.; Wang, Y.; Zhu, Q.; Mao, J.; Xiao, C.; Lu, X.; Zhang, H. A Surface Defect Detection Framework for Glass Bottle Bottom Using Visual Attention Model and Wavelet Transform. *IEEE Trans. Ind. Inform.* **2019**, *16*, 2189–2201. [[CrossRef](#)]
27. Mortin, K.; Golovko, Y.; Pogozheva, A.; Rybakov, A. Development of Hardware-Software Complex of Vision System for Detection of Sheet Metal Rolling Defects on Lighted and Noisy Images. In *Proceedings of the 2023 7th International Conference on Information, Control, and Communication Technologies (ICCT)*; IEEE: Piscataway, NJ, USA, 2023; pp. 1–7.
28. Priyanka, V.; Rama, Y.S.; Sravani, K.; Kavya, B. Implementation of Sobel Edge Detection with Image Processing on FPGA. In *Proceedings of the 2024 2nd World Conference on Communication & Computing (WCONF)*; IEEE: Piscataway, NJ, USA, 2024; pp. 1–5.
29. Huang, H.; Luo, J. Research on the Implementation and Application of Improved Canny Algorithm for Image Edge Detection Based on FPGA. In *Proceedings of the International Conference on Image Processing, Machine Learning and Pattern Recognition*; SPIE: Bellingham, WA, USA, 2024; pp. 55–61.
30. Abdullah, H.A.T.; Mahmood, R.Z.; Alhaj Zber, S.M.; A.Mohammed, R.; Ahmed, M.R.; Talab, A.W. FPGA-Based Three Edge Detection Algorithms (Sobel, Prewitt and Roberts) Implementation for Image Processing. *Przegl. Elektrotech.* **2024**, *2*, 29–33. [[CrossRef](#)]
31. Liu, Y.; Yao, J.; Lu, X.; Xie, R.; Li, L. DeepCrack: A Deep Hierarchical Feature Learning Architecture for Crack Segmentation. *Neurocomputing* **2019**, *338*, 139–153. [[CrossRef](#)]
32. Cao, W.; Liu, Q.; He, Z. Review of Pavement Defect Detection Methods. *IEEE Access* **2020**, *8*, 14531–14544. [[CrossRef](#)]
33. Nguyen, S.D.; Tran, T.S.; Tran, V.P.; Lee, H.J.; Piran, J.; Le, V.P. Deep Learning-Based Crack Detection: A Survey. *Int. J. Pavement Res. Technol.* **2023**, *16*, 943–967. [[CrossRef](#)]
34. Joshi, A.; Mehta, A. Mammogram Image Classification Using Various Machine Learning Algorithms. In *Proceedings of the 2022 7th International Conference on Computing, Communication and Security (ICCCS)*; IEEE: Seoul, Republic of Korea, 2022. [[CrossRef](#)]
35. Liu, H.; Jia, X.; Su, C.; Yang, H.; Li, C. Tire Appearance Defect Detection Method via Combining HOG and LBP Features. *Front. Phys.* **2023**, *10*, 1099261. [[CrossRef](#)]
36. Madhumitha, S.; Narayana, S.K.S.; Dhevi, S.Y.; Venkataraman, M.; Nadarajah, S. A Hybrid Machine Learning Framework for Image-Based Bearing Fault Diagnosis Advancing SDG Goal 9: Sustainable Industry and Innovation. *Array* **2026**, *31*, 101017. [[CrossRef](#)]
37. Gyimah, N.K.; Girma, A.; Mahmoud, M.N.; Nateghi, S.; Homaifar, A.; Opoku, D. A Robust Completed Local Binary Pattern (RCLBP) for Surface Defect Detection. In *Proceedings of the 2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*; IEEE: Melbourne, Australia, 2021; pp. 1927–1934. [[CrossRef](#)]
38. Juneja, R.; Dudy, A. Optimized Classification of Steel Surface Defects via Hybrid Features and Neighborhood Component Analysis. *Int. J. Image Graph. Signal Process.* **2026**, *18*, 71. [[CrossRef](#)]
39. Ghilardi, M.C.; Jacques, J.C.; Manssour, I. Crosswalk Localization from Low Resolution Satellite Images to Assist Visually Impaired People. *IEEE Comput. Graph. Appl.* **2016**, *38*, 30–46. [[CrossRef](#)] [[PubMed](#)]
40. Votyakov, E.V.; Andreou, A.; Liarakapis, F. An Augmented Reality System with an Offline LSTM-Based Fault Recognition Model for Sewer Pipeline Inspection. *IEEE Access* **2026**, *14*, 19862–19876. [[CrossRef](#)]
41. Pemila, M.; Pongiannan, R.K.; Narayanamoorthi, R.; Sweelem, E.A.; Hendawi, E.; Abu El-Sebah, M.I. Real-Time Classification of Vehicles Using Machine Learning Algorithm on the Extensive Dataset. *IEEE Access* **2024**, *12*, 98338–98351. [[CrossRef](#)]
42. Anowar, F.; Sadaoui, S.; Selim, B. Conceptual and Empirical Comparison of Dimensionality Reduction Algorithms (PCA, KPCA, LDA, MDS, SVD, LLE, Isomap, LE, ICA, t-SNE). *Comput. Sci. Rev.* **2021**, *40*, 100378. [[CrossRef](#)]
43. Anjum, R.; Dipti, R.R.; Rashid, H.O.; Ripon, S. An Efficient Breast Cancer Analysis Technique by Using a Combination of HOG and Canny Edge Detection Techniques. In *Proceedings of the 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*; IEEE: Tirunelveli, India, 2021; pp. 1290–1295. [[CrossRef](#)]
44. Liu, Q.; Song, Y.; Tang, Q.; Bu, X.; Hanajima, N. Wire Rope Defect Identification Based on ISCM-LBP and GLCM Features. *Vis. Comput.* **2024**, *40*, 545–557. [[CrossRef](#)]
45. Li, G.; Yu, Z.; Yang, K.; Lin, M.; Chen, C.L.P. Exploring Feature Selection with Limited Labels: A Comprehensive Survey of Semi-Supervised and Unsupervised Approaches. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 6124–6144. [[CrossRef](#)]

46. Zhang, F.; Li, W.; Zhang, Y.; Feng, Z. Data Driven Feature Selection for Machine Learning Algorithms in Computer Vision. *IEEE Internet Things J.* **2018**, *5*, 4262–4272. [\[CrossRef\]](#)
47. Ciyilan, B.; Polat, H.; Akyel, C. Öznitelik Seçimi ile Desteklenen Makine Öğrenmesine Dayalı Göğüs Kanserinin Erken Tespiti ve Teşhisi. *Gazi Univ. J. Sci. Part C Des. Technol.* **2024**, *12*, 675–690. [\[CrossRef\]](#)
48. Öztekin, Y.; Gadalla, O. Image Processing and Traditional Machine Learning Based Classification of Brown Marmorated Stink Bug (*Halyomorpha halys*) Defected Hazelnut. *Tekirdağ Ziraat Fak. Derg.* **2023**, *20*, 784–798. [\[CrossRef\]](#)
49. Silva, E.T., Jr.; Sampaio, F.; da Silva, L.C.; Medeiros, D.S.; Correia, G.P. A Method for Embedding a Computer Vision Application into a Wearable Device. *Microprocess. Microsyst.* **2020**, *76*, 103086. [\[CrossRef\]](#)
50. Khaire, U.M.; Dhanalakshmi, R. Stability of Feature Selection Algorithm: A Review. *J. King Saud Univ.-Comput. Inf. Sci.* **2022**, *34*, 1060–1073. [\[CrossRef\]](#)
51. Jin, J.; Xiao, R.; Daly, I.; Miao, Y.; Wang, X.; Cichocki, A. Internal Feature Selection Method of CSP Based on L1-Norm and Dempster–Shafer Theory. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4814–4825. [\[CrossRef\]](#) [\[PubMed\]](#)
52. Cao, Y.; Sun, Y.; Li, P.; Su, S. Vibration-Based Fault Diagnosis for Railway Point Machines Using Multi-Domain Features, Ensemble Feature Selection and SVM. *IEEE Trans. Veh. Technol.* **2023**, *73*, 176–184. [\[CrossRef\]](#)
53. Prunella, M.; Scardigno, R.M.; Buongiorno, D.; Brunetti, A.; Longo, N.; Carli, R.; Dotoli, M.; Bevilacqua, V. Deep Learning for Automatic Vision-Based Recognition of Industrial Surface Defects: A Survey. *IEEE Access* **2023**, *11*, 43370–43423. [\[CrossRef\]](#)
54. Khanam, R.; Hussain, M.; Hill, R.; Allen, P. A Comprehensive Review of Convolutional Neural Networks for Defect Detection in Industrial Applications. *IEEE Access* **2024**, *12*, 94250–94295. [\[CrossRef\]](#)
55. Yang, K.; Liu, Y.; Zhang, S.; Cao, J. Surface Defect Detection of Heat Sink Based on Lightweight Fully Convolutional Network. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–12. [\[CrossRef\]](#)
56. Zhang, T.; Qin, L.; Zou, Q.; Zhang, L.; Wang, R.; Zhang, H. Crackscopenet: A Lightweight Neural Network for Rapid Crack Detection on Resource-Constrained Drone Platforms. *Drones* **2024**, *8*, 417. [\[CrossRef\]](#)
57. Moosmann, J.; Müller, H.; Zimmerman, N.; Rutishauser, G.; Benini, L.; Magno, M. Flexible and Fully Quantized Lightweight Tinyissimoyolo for Ultra-Low-Power Edge Systems. *IEEE Access* **2024**, *12*, 75093–75107. [\[CrossRef\]](#)
58. Hong, S.; Park, G.; Kim, J.S. Automated Deep-Learning Model Optimization Framework for Microcontrollers. *ETRI J.* **2025**, *47*, 179–192. [\[CrossRef\]](#)
59. Kwon, J.; Park, D. Efficient Partial Weight Update Techniques for Lightweight On-Device Learning on Tiny Flash-Embedded MCUs. *IEEE Embed. Syst. Lett.* **2023**, *15*, 206–209. [\[CrossRef\]](#)
60. Orășan, I.L.; Bublea, A.I.; Căleanu, C.D. Deep Learning-Based Eye Gaze Estimation for Automotive Applications Using Knowledge Distillation. *IEEE Access* **2023**, *11*, 120741–120753. [\[CrossRef\]](#)
61. Novac, P.E.; Boukli Hacene, G.; Pegatoquet, A.; Miramond, B.; Gripon, V. Quantization and Deployment of Deep Neural Networks on Microcontrollers. *Sensors* **2021**, *21*, 2984. [\[CrossRef\]](#) [\[PubMed\]](#)
62. Maciá-Lillo, A.; Barrachina, S.; Fabregat, G.; Dolz, M.F. Optimizing Convolutions for Deep Learning Inference on ARM Cortex-M Processors. *IEEE Internet Things J.* **2024**, *11*, 26203–26219. [\[CrossRef\]](#)
63. Wang, X.; Liang, Y.; Yang, X.; Guo, M.; Jia, T.; Kamran, M. A Real-Time Multiscale Pavement Crack-Detection Model. *Signal Image Video Process.* **2025**, *19*, 1237. [\[CrossRef\]](#)
64. Li, H.; Peng, T.; Qiao, N.; Guan, Z.; Feng, X.; Guo, P.; Duan, T.; Gong, J. CrackTinyNet: A Novel Deep Learning Model Specifically Designed for Superior Performance in Tiny Road Surface Crack Detection. *IET Intell. Transp. Syst.* **2024**, *18*, 2693–2712. [\[CrossRef\]](#)
65. Geng, S.; Zhu, Z.; Wang, Z.; Dan, Y.; Li, H. LW-ViT: The Lightweight Vision Transformer Model Applied in Offline Handwritten Chinese Character Recognition. *Electronics* **2023**, *12*, 1693. [\[CrossRef\]](#)
66. Luo, G.; Zhou, Y.; Sun, X.; Wang, Y.; Cao, L.; Wu, Y.; Huang, F.; Ji, R. Towards Lightweight Transformer via Group-Wise Transformation for Vision-and-Language Tasks. *IEEE Trans. Image Process.* **2022**, *31*, 3386–3398. [\[CrossRef\]](#) [\[PubMed\]](#)
67. Zhu, Y.; Xia, Q.; Jin, W. SRDD: A Lightweight End-to-End Object Detection with Transformer. *Connect. Sci.* **2022**, *34*, 2448–2465. [\[CrossRef\]](#)
68. Melikyan, V.; Khachatryan, T.; Galstyan, D.; Harutyunyan, E. Efficient Vision Transformer Deployment on Google Coral Through Knowledge Distillation. In *Proceedings of the 2024 IEEE East-West Design & Test Symposium (EWDTS)*; IEEE: Yerevan, Armenia, 2024; pp. 1–3. [\[CrossRef\]](#)
69. Yang, G.; Yan, X.; Kou, H.; Zou, Z.; Wei, Q.; Cai, H.; Liu, B. TWDP: A Vision Transformer Accelerator with Token-Weight Dual-Pruning Strategy for Edge Device Deployment. In *Proceedings of the 30th Asia and South Pacific Design Automation Conference (ASP-DAC)*; ACM: Tokyo, Japan, 2025; pp. 177–182. [\[CrossRef\]](#)
70. Misawa, N.; Wang, T.; Matsui, C.; Takeuchi, K. Transformer Hetero-CiM: Heterogeneous Integration of ReRAM CiM and SRAM CiM for Vision Transformer at Edge Devices. In *Proceedings of the 30th Asia and South Pacific Design Automation Conference (ASP-DAC)*; ACM: Tokyo, Japan, 2025; pp. 332–334. [\[CrossRef\]](#)
71. Khadka, R.; Sengupta, P.; Lind, G.P.; Yazidi, A. Quantization of Vision Transformer-Based Model for Real-Time EEG Classification. In *Nordic E-Infrastructure Collaboration Conference (NeIC)*; Springer Nature: Cham, Switzerland, 2024; pp. 17–27.

72. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: Las Vegas, NV, USA, 2016; pp. 770–778. [[CrossRef](#)]
73. Seol, D.; Kim, H.; Park, S.; Lee, J. Comparative Study of CNN-Based Deep Learning Models for Concrete Crack Detection. *J. Build. Eng.* **2025**, *112*, 113651. [[CrossRef](#)]
74. Liu, Y.; Xue, J.; Li, D.; Zhang, W.; Chiew, T.K.; Xu, Z. Image Recognition Based on Lightweight Convolutional Neural Network: Recent Advances. *Image Vis. Comput.* **2024**, *146*, 105037. [[CrossRef](#)]
75. Howard, A.; Sandler, M.; Chen, B.; Wang, W.; Chen, L.-C.; Tan, M.; Chu, G.; Vasudevan, V.; Zhu, Y.; Pang, R.; et al. Searching for MobileNetV3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*; IEEE: Seoul, Republic of Korea, 2019; pp. 1314–1324. [[CrossRef](#)]
76. 360DigiTMG. Egg Crack Detection Dataset. Roboflow Universe. Available online: <https://universe.roboflow.com/360digitmg-qf6sf/egg-crack-detection-ba5lc> (accessed on 21 January 2026).
77. ProjecProject-Creo0. Crack Dataset. Roboflow Universe. Available online: <https://universe.roboflow.com/project-creo0/crack-vudec> (accessed on 22 January 2026).
78. Omoebamije, O. Concrete & Pavement Crack Dataset. Kaggle. Available online: <https://www.kaggle.com/datasets/oluwaseunad/concrete-and-pavement-crack-images> (accessed on 28 July 2026).
79. Buerhop-Lutz, C.; Deitsch, S.; Maier, A.; Gallwitz, F.; Berger, S.; Doll, B.; Hauch, J.A.; Camus, C.; Brabec, C.J. A Benchmark for Visual Identification of Defective Solar Cells in Electroluminescence Imagery. In *Proceedings of the 35th European PV Solar Energy Conference and Exhibition (EU PVSEC)*; WIP: Minneapolis, MN, USA, 2018; pp. 1287–1289. [[CrossRef](#)]
80. Topi, M.; Timo, O.; Matti, P.; Maricor, S. Robust Texture Classification by Subsets of Local Binary Patterns. In *Proceedings of the 15th International Conference on Pattern Recognition (ICPR)*; IEEE: Barcelona, Spain, 2000; pp. 935–938. [[CrossRef](#)]
81. Twum, F.; Missah, Y.M.; Oppong, S.O.; Ussiph, N. Textural Analysis for Medicinal Plants Identification Using Log Gabor Filters. *IEEE Access* **2022**, *10*, 83204–83220. [[CrossRef](#)]
82. Zhang, D. Wavelet Transform. In *Fundamentals of Image Data Mining: Analysis, Features, Classification and Retrieval*; Springer International Publishing: Cham, Switzerland, 2019; pp. 35–44. [[CrossRef](#)]
83. Bishop, C.M.; Nasrabadi, N.M. *Pattern Recognition and Machine Learning*; Springer: New York, NY, USA, 2006; Volume 4, p. 738.
84. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: Salt Lake City, UT, USA, 2018; pp. 4510–4520. [[CrossRef](#)]
85. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556. [[CrossRef](#)]
86. Tan, M.; Le, Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*; PMLR: Cambridge, MA, USA, 2019; pp. 6105–6114.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.