electronics

MDPI

Article

Textile Defect Detection Using Artificial Intelligence and Computer Vision—A Preliminary Deep Learning Approach

Rúben Machado [1,*], Luis A. M. Barros [1], Vasco Vieira [1], Flávio Dias da Silva [1], Hugo Costa [1] and Vitor Carvalho [2]

[1] CeNTI—Centre for Nanotechnology and Advanced Materials, 4760-034 Vila Nova de Famalicão, Portugal; lbarros@centi.pt (L.A.M.B.); vvieira@centi.pt (V.V.); fdsilva@centi.pt (F.D.d.S.); hcosta@centi.pt (H.C.)
[2] 2Ai, School of Technology, Polytechnic University of Cávado and Ave, 4750-810 Barcelos, Portugal; vcarvalho@ipca.pt
* Correspondence: rmachado@centi.pt

Abstract

Fabric defect detection is essential for quality assurance in textile manufacturing, where manual inspection is inefficient and error-prone. This paper presents a real-time deep learning-based system leveraging YOLOv11 for detecting defects such as holes, color bleeding and creases on solid-colored, patternless cotton and linen fabrics using edge computing. The system runs on an NVIDIA Jetson Orin Nano platform and supports real-time inference, Message Queuing Telemetry (MQTT)-based defect reporting, and optional Real-Time Messaging Protocol (RTMP) video streaming or local recording storage. Each detected defect is logged with class, confidence score, location and unique ID in a Comma Separated Values (CSV) file for further analysis. The proposed solution operates with two RealSense cameras placed approximately 1 m from the fabric under controlled lighting conditions, tested in a real industrial setting. The system achieves a mean Average Precision (mAP@0.5) exceeding 82% across multiple synchronized video sources while maintaining low latency and consistent performance. The architecture is designed to be modular and scalable, supporting plug-and-play deployment in industrial environments. Its flexibility in integrating different camera sources, deep learning models, and output configurations makes it a robust platform for further enhancements, such as adaptive learning mechanisms, real-time alerts, or integration with Manufacturing Execution System/Enterprise Resource Planning (MES/ERP) pipelines. This approach advances automated textile inspection and reduces dependency on manual processes.

Keywords: computer vision; deep learning; edge computing; fabric defect detection

check for updates

Academic Editor: Yang Liu and Jin Li

Received: 17 July 2025
Revised: 2 September 2025
Accepted: 9 September 2025
Published: 18 September 2025

Citation: Machado, R.; Barros, L.A.M.; Vieira, V.; Silva, F.D.d.; Costa, H.; Carvalho, V. Textile Defect Detection Using Artificial Intelligence and Computer Vision—A Preliminary Deep Learning Approach. *Electronics* **2025**, *14*, 3692. https://doi.org/10.3390/electronics14183692

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The textile industry is one of the most significant industries in Portugal, accounting for approximately 8 % of the country's total exports [1]. The Portuguese textile industry has undergone substantial modernization and innovation, with a focus on producing high-quality, value-added products. However, the growing demand for fast fashion has led to a corresponding increase in the generation of textile waste. Among the various consequences of this accelerated and globalized production model are the severe environmental impacts associated with the large volumes of textile waste that are discarded due to manufacturing defects. Despite the amount of textile waste sent to landfills decreasing from 21% to 12% in Europe between 2010 and 2022, incineration increased from 10% to 14% during the same

period. The amount of textile waste generated by EU member states is approximately 6.94 million tonnes, which represents 16 kg per person [2].

Preventive measures, particularly the early detection of textile defects during the initial stages of the production process, play a crucial role in reducing environmental impact by minimizing textile waste and enhancing economic efficiency by lowering operation and production costs associated with undetected defects. The predominant method for textile inspection remains traditional visual assessment performed by experienced inspectors. However, this approach is inherently inconsistent and susceptible to errors due to human limitations such as fatigue and visual strain. Moreover, it involves significant operational costs, which can be reduced through the implementation of automated inspection systems. It is estimated that the price of fabrics can be negatively affected by 45–64% due to the presence of defects [3]. It has also been estimated that the defect detection efficiency of traditional visual inspection is approximately 75 %, with performance declining further under conditions of increased fatigue and strain [4].

Initially, to improve the accuracy and reliability of fabric defect detection, several studies have been conducted [4–7], primarily employing statistical analysis [8–11], spectral analysis [12–15], and model-based analysis [5–8]. Nevertheless, these conventional methods often encounter limitations related to low noise tolerance, limited adaptability to diverse fabric textures, and constraints in real-time applicability.

In parallel with advances in hardware processing capabilities, recent innovations have leveraged deep learning architectures, particularly convolutional neural networks (CNNs), to overcome the limitations of traditional methods. More specifically, in the domain of object detection, models such as You Only Look Once (YOLO) [9], Single Shot Detector (SSD) [10], and Faster R-CNN [11], among other classic approaches, have been successfully applied in various contexts, including intruder detection in home systems [12] and disease identification in plant leaves [13].

In the context of automatic fabric defect detection within industrial environments, key challenges include noise interference, system processing speed, the quality of the training dataset, and model generalization. Data acquired in complex industrial settings are highly susceptible to external variables such as lighting variations and background noise, which can significantly degrade the detection accuracy of machine learning (ML) models [14]. Consequently, constructing a high-quality and representative dataset is essential for achieving reliable performance. Recent research suggests that data augmentation techniques can enhance dataset diversity, thereby improving the robustness and generation capability of deep learning models [15].

This paper proposes an automatic system for detecting holes, color bleeding, and crease defects on plain solid multicolor, patternless cotton and linen textiles, integrating computer vision and deep learning into an edge-computing platform capable of real-time defect identification. The system supports wireless visualization of operational status and records all detected defects for post-processing and analysis. The objective is to develop a low-power, adaptable, non-invasive, autonomous solution that facilitates the adoption of automatic quality control in already existing equipment used in textile manufacturing, thereby minimizing reliance on manual inspection processes while maintaining high standards with minimal investment. This system was implemented in an industrial environment, and the obtained results are presented and analyzed. The main contributions are listed as follows:

1. This paper introduces an automatic fabric defect detection system that integrates a lightweight framework, such as Ultralytics' YOLO, with an edge device like the NVIDIA Jetson Orin Nano, aiming to reduce processing latency and enhance production line efficiency.

2. A robust dataset consisting of images taken from fabrics with different colors is used, contributing to a more generalized model.
3. Advanced data augmentation techniques are employed to train a more robust model capable of generalizing across diverse industrial environments.
4. In order to accommodate the constrained resources of edge devices, the trained model is optimized using TensorRT, ensuring compliance with real-time performance requirements.

Collectively, these contributions constitute a scalable and efficient automatic defect detection system suitable for a wide range of industrial applications.

The remainder of this paper is organized as follows: Section 2 reviews related work; Section 3 details the architecture and implementation of the proposed system; Section 4 presents and analyzes the experimental results; Section 5 discusses the results obtained; and Section 6 finishes the paper with conclusions and future work.

2. State of the Art

In general, automatic fabric defect detection relies on a structured pipeline aimed at replacing manual visual inspection with intelligent systems that offer enhanced speed, accuracy and consistency. The core architecture of such systems typically comprises four primary stages: (i) image acquisition; (ii) pre-processing; (iii) feature extraction; and (iv) classification. These stages collectively guide the entire workflow, from image capture to final defect classification, as illustrated in Figure 1.

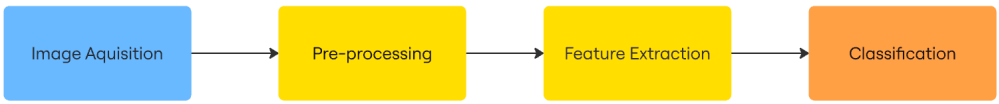


Figure 1. Fabric defect detection general architecture [16].

This processing flow is a foundational component of both traditional and artificial intelligence (AI)-based fabric defect detection systems [16]. In recent years, the majority of research efforts have concentrated on advanced algorithms that leverage the advantages of deep learning, particularly in terms of feature extraction, processing speed and deployment feasibility on resource-constrained systems [4]. Nevertheless, earlier studies have explored the application of conventional image processing techniques for fabric defect detection, providing valuable insights into the evolution of the field [17]. In this section, the main concepts, actions and algorithms to be integrated into each block are analyzed.

2.1. Image Acquisition

The first stage is responsible for acquiring the image that will feed the entire control algorithm. Therefore, implementing a precise and robust system requires a high-quality acquisition setup to reduce the need for extensive post-processing. This setup, in addition to providing adequate lighting, must include a camera with high image resolution, contrast, and consistent illumination to ensure continuous and reliable defect detection, even for the most subtle flaws.

From a mechanical system perspective, the following components can be identified: (i) cameras—responsible for image capture; (ii) lighting—ensures uniform illumination and highlights defects; (iii) optics—adjusts focus and decouples the field of view from depth of field; (iv) mechanical motion systems—responsible for linear movement and continuous fabric inspection.

Regarding cameras, there are different types of technologies available on the market. For example, area-scan cameras capture two-dimensional (2D) images similarly to a standard photographic camera. Authors of [18] described a low-cost, real-time defect inspection

system using an area-scan camera coupled with MATLAB processing. Their study showed this configuration was effective for static or slow-moving items but highlighted limitations in the field of view and throughput in high-speed industrial settings. In terms of defect detection, they are capable of identifying stains, lines, pattern inconsistencies, and loose threads. Despite being low-cost and ideal for prototypes, they have a limited field of view and relatively low efficiency.

Line-scan cameras capture one line of pixels at a time and are particularly suited for detecting surfaces in continuous motion. Authors of [19] developed an automated optical inspection (AOI) system using line-scan cameras for fabric inspection and reported high spatial resolution and suitability for fast-moving rolls of fabric. However, they noted challenges with illumination uniformity and the need for precise synchronization with fabric movement to avoid distortion or missed defects. This type of camera is ideal for high-speed, continuous production lines. In terms of defect detection, they enable the identification of broken threads, transverse lines, fine stains, and contrast points (both dark and light). While they are well-suited for large, continuous surfaces and offer high-resolution imaging, their use requires precise synchronization with the fabric's movement as well as consistent and uniform lighting.

Multispectral cameras are capable of capturing images across different bands of light, such as RGB, infrared, and ultraviolet. Authors of [20] applied multispectral imaging to Jacquard fabrics and showed enhanced detection of both visible and invisible defects. They emphasized its strength in detecting chemical stains and yarn inconsistencies that evade Red, Green and Blue (RGB)-only methods. However, they acknowledged the computational overhead and strict requirements for controlled lighting conditions as major constraints. These cameras are suitable for identifying defects that are not visible to the human eye, including chemical stains, invisible contaminations, variations in yarn composition, or inconsistencies in dye absorption. Although they offer the advantage of detecting both textural and chemical anomalies, they require constant, controlled lighting and significant processing power to operate effectively.

Red, Green, Blue, and Depth (RGB-D) cameras enable the simultaneous capture of RGB images along with depth maps, utilizing infrared sensors and stereoscopic techniques. Authors of [21] utilized RGB-D data for three-dimensional (3D) textile inspection, focusing on volumetric and structural anomalies. Their work showed that depth data helps distinguish between shadows and real defects, but the need for depth normalization, high processing cost, and complex camera calibration are noted as limitations. These cameras are particularly suited for identifying three-dimensional defects such as wrinkles, folds, creases, surface deformations, and texture irregularities. They offer advantages such as low acquisition cost, real-time capture capability, and good portability. However, their precision is generally lower compared to other industrial imaging systems, and their performance tends to decline on reflective or very smooth surfaces, such as silk or satin.

Lastly, thermal cameras capture the heat emissions from objects. As such, they are best suited for identifying thermal-related defects, such as overheating, uneven heat dissipation, or friction. The author of [22] proposed a novel thermal defect detection method for fabric surfaces. They demonstrated success in identifying defects invisible to visual spectrum imaging, such as yarn density variation. However, they reported high sensitivity to ambient temperature changes and thermal noise, which can lead to false positives without environmental control. Although they enable the detection of defects that are not visible through color or texture analysis, they are highly sensitive to thermal noise and require controlled environmental conditions to ensure accurate readings.

2.2. Pre-Processing

Pre-processing constitutes a complementary stage to image acquisition, aiming to improve the quality and consistency of the captured images [23,24]. By employing various enhancement techniques, it is possible to reduce undesirable variations, highlight relevant features, and normalize the image. In scenarios involving uneven illumination, reflections, or image noise, minimizing these inconsistencies is essential to prevent them from being misinterpreted as defects. The authors of [25] addressed uneven lighting by applying global gamma correction and local MS-BOX filtering, followed by saliency enhancement. This approach improved contrast and feature clarity but risked over-enhancement, which could obscure genuine defects or introduce false ones. Feature enhancement facilitates the identification of textures, edges, or fabric structures, thereby contributing to more accurate defect detection.

The authors of [5] utilized saliency mapping to highlight defect-prone regions under varying illumination conditions. Their technique effectively reduced background-induced false positives but also had limitations—it could diminish low-contrast defects that lacked strong saliency cues. Image normalization further increases the robustness of detection algorithms by enabling better generalization across varying acquisition conditions.

The authors of [26] employed adaptive thresholding and filtering to identify conductive anomalies in smart textiles. Their results showed high accuracy, but they emphasized that threshold parameters must be tuned to the specific fabric type and lighting setup to avoid false positives from benign texture deviations. However, care must be taken during this process to avoid suppressing genuine defects or exaggerating normal texture patterns, which could result in false negatives or false positives, respectively. Authors across multiple studies have reported that while preprocessing enhances system reliability, it must be calibrated with domain knowledge to avoid overfitting to lighting or texture idiosyncrasies [5,25].

The authors of [27] propose a method using the Inverse Discrete Fourier Transform that removes most of the periodic patterns, returning a uniform image where only defective areas are preserved. They also apply wavelet shrinkage to denoise the returned image.

Gaussian filtering is a very common practice in image noise reduction. It works by smoothing an image through averaging neighbouring pixels with weights from a Gaussian distribution, thus reducing noise and detail. If applied in excess, it can have downsides like vanishing very small defects. The authors of [28] indicate that noise is generated due to factors such as environment or signal noise, so they use it to filter out that noise, avoiding misdetections.

Sobel filtering, on the other hand, is commonly used for edge detection by utilizing two 3×3 convolution kernels to calculate the gradient at each pixel in horizontal and vertical directions. The magnitude and direction of the gradient are then used to determine the presence and orientation of edges. The authors of [29] propose a method of fabric defect detection where a Sobel Operator is used to detect the edges of each image, obtaining a coarse binary image. Using this technique alone generated too many false alarms, so they applied a Patch Statistics Algorithm to reduce defect noise.

Histogram equalization enables the normalization of brightness intensity across the entire image, making dark areas more visible and bright regions more distinct. This enhancement improves the visibility of hidden details, increases contrast between different regions, and emphasizes textures. As a result, it proves to be a useful technique for detecting holes, tears, and creases, as it enhances contrast along irregular edges and highlights shadows between distinct surfaces. Authors of [30,31] implemented histogram equalization as a crucial preprocessing step, employing it on grayscale images before segmentation and neural classification. They observed improved feature separation but

also highlighted that the equalized grayscale images could not capture chromatic nuances, which are essential for identifying certain textile defects. Since discoloration is a color-based defect rather than one based on contrast or intensity, histogram equalization may suppress these chromatic variations, making it unsuitable for detecting such defects.

Image data augmentation is a widely adopted technique to expand limited datasets and improve generalization by artificially expanding the diversity of data. This approach reduces overfitting and improves robustness against common challenges in textile datasets, including uneven illumination and noisy environments. Empirical studies demonstrate that models trained using data augmentation often outperform those relying on explicit regularization techniques [32,33]. Data augmentation techniques include geometric modifications (rotation, flipping, cropping), color and intensity adjustments (brightness, contrast, exposure), noise injection, as well as advanced mix-based methods like MixUp, CutMix, and CutOut [34].

The authors of [33] compare three different augmentation sets, composed of simple transformations such as flipping, exposure, blurring, and mosaicking. Results show that augmentation techniques have a positive impact on the model result, with a higher mean Average Precision(mAP) while using all the mentioned techniques. However, they observed that the usage of blur can reduce precision and thus emphasize the need to choose augmentation methods carefully. Corroborating this, ref. [34] notes that there is a lack of data augmentation approaches tailored to industrial settings, where most defect detection tasks still rely on conventional augmentation techniques.

2.3. Feature Extraction

Feature extraction is a critical step in the fabric defect detection pipeline, as it aims to identify and quantify relevant patterns from the image that allow for reliable differentiation between defective and non-defective fabrics. This stage transforms raw visual data into a set of descriptive attributes that serve as input for the classification module [35,36].

Among classical paradigms, Gabor filters are widely employed for enhancing texture patterns across multiple scales and orientations. The authors of [37] apply Gabor filters effectively by eliminating texture interference, enhancing the model's ability to detect various fabric defects, including tiny defects and those in uneven fabrics with creases. In combination with GA for parameter tuning, they achieved an improvement of approximately 16% compared to the same vanilla model.

In more recent approaches, feature extraction is performed implicitly within deep learning models—especially CNNs—which learn optimal filters directly from labeled data. In this CNN, early convolutional layers learn to detect basic visual elements like edges, textures, and corners, and as the network deepens, intermediate layers combine these simple patterns into more abstract representations of texture motifs and structural anomalies. Activation functions [38] (e.g., Rectified Linear Unit(ReLU) [39]), batch normalization [40], and pooling layers [41] further refine and stabilize these learned representations, emphasizing the most relevant features while reducing overfitting.

Regardless of the technique, the quality and discriminative power of extracted features are pivotal to the robustness and generalizability of the defect detection system, particularly under variable lighting, textures, and fabric movement.

2.4. Classification

The classification stage is the final decision-making process, where extracted features are analyzed to determine whether a defect is present, and if so, to identify its type. This step is critical, as it defines the overall performance of the system in terms of accuracy, sensitivity, and reliability.

Although limited, the literature includes some comparative analyses of different algorithms applied to textile defect detection. In [42], a comparative study is conducted involving Faster R-CNN, Cascade R-CNN, and nine versions of the YOLO algorithm. The study reports performance metrics such as mean Average Precision at IoU (Intersection over Union) threshold 0.5 (mAP@0.5), frames per second (FPS), and Giga Floating Point Operations per Second (GFLOPS). The results indicate that YOLO-based models outperform Faster R-CNN and Cascade R-CNN in terms of both detection accuracy and processing efficiency, with consistent improvements observed across successive YOLO versions. Specifically, YOLOv5s achieves the highest acquisition speed (590.88 FPS), while YOLOv11s demonstrates superior computational efficiency (21.3 GFLOPS). The proposed enhanced version of YOLOv8s attains the highest detection accuracy, achieving 0.446 mAP@0.5.

In addition to defect detection accuracy, performance in real-time systems is equally critical. The following sections provide a more detailed analysis of the different algorithms under consideration.

i. Traditional classifiers

The term traditional classifiers in this field refers to statistical and shallow Machine Learning (ML) models [43] that assign a label to the input image or vector [28]. Common examples include Support Vector Machines (SVM), k-Nearest Neighbors (KNN), and Decision Trees, which have been commonly used in systems based on handcrafted features. Authors of [44] applied a hybrid approach using Histogram of Oriented Gradients (HOG) features with SVM classification to detect fabric defects in grayscale images. Their results showed high classification accuracy with strong generalization on small datasets. However, they emphasized that SVM performance was sensitive to feature scaling and suffered under inconsistent lighting and complex background textures, necessitating extensive preprocessing to remain reliable. In the case of Decision Tree-based applications, the authors of [45] applied a Decision Tree-based Random Forest classifier on Gabor and Gray Level Co-Occurrence Matrix (GLCM) features. While the model effectively handled class diversity, it noted increased computational cost during training and a degradation in accuracy under noisy and unstructured textile backgrounds. In another example of KNN algorithm applications, the authors of [46] implemented a KNN classifier using multi-scale contour features. Their system reached a high classification accuracy (95.48%) and outperformed several alternatives. However, it also misclassified non-defective fabrics with high edge curvature, demonstrating vulnerability to subtle shape distortions and overlapping features among classes. Some other models rely heavily on hand-crafted low-level features, presenting issues with adaptability and generalization for variations in the defect's classification [47].

Those previous classifier models, instead of learning automatically, require engineered features (e.g., HOG) [48], which necessitate a significant effort in complex visual analyses involving numerous variants and patterns. Another defect is that these methods lack recognition of spatial relations in the image, such as certain shapes, contexts, and textural continuity [49]. In terms of scalability, these models also present some issues; for example, the SVM can require a vast amount of CPU usage, which makes those classifiers impractical for edge deployment.

In recent years, other systems have been developed, such as deep learning object detectors (CNN-based detectors), obtaining better results [50], because of their ability to detect and localize instances of objects in images, bounding boxes and classes for each object. The advent of CNNs has significantly enhanced the classification process by combining feature extraction and classification into a unified framework. These models can detect highly subtle and nonlinear patterns, making them ideal for industrial scenarios. Modern architectures like YOLO, SSD, and Faster R-CNN have been successfully applied

to classify and localize multiple defect types in real time, offering both high accuracy and operational speed.

ii. Faster R-CNN:

Faster R-CNN is a two-stage detection algorithm that employs a Region Proposal Network (RPN) to generate object bounding boxes [11]. At the first stage, a selective search algorithm is applied to perform large-scale region extraction. The method also integrates SVMs for classification, which significantly accelerates detection speed [51]. Under these conditions, such models can achieve an Average Precision of 42.7% and an inference speed of 17 fps [50].

Due to its computational capacity, this model adapts efficiently to the analysis of various fabric types through its hybrid approach, which combines filtering and statistical approximations to improve accuracy [51]. This characteristic is particularly advantageous for detecting small flaws, as the dataset can be oriented toward small targets.

Authors of [37] proposed an enhanced Faster R-CNN model incorporating Gabor filters and a Genetic Algorithm (GA) optimization strategy. The term Genetic Algorithm Faster R-CNN (Faster R-CNN) achieved a mean average precision (mAP) of 94.57%, outperforming vanilla Faster R-CNN (78.98%) by better suppressing texture noise and improving localization. Despite its accuracy gains, the model incurred higher computational demands due to the additional Gabor preprocessing and GA training stages, potentially limiting its suitability for real-time or edge applications.

Nevertheless, due to its high computational requirements, the model suffers from reduced inference speed, which limits its applicability in real-time scenarios. This issue is better addressed by single-stage detectors such as SSD and YOLO [50]. Another disadvantage is the necessity of defining numerous parameters, like learning rate and loss rate, to optimize the algorithm, which increases both the time and expertise required for deployment, training, or model updates [52].

iii. SSD (Single Shot MultiBox Detector):

The SSD algorithm, proposed by Liu et al. [10], is based on the principle of single-shot multibox detection. Specifically, this method relies on multi-scale feature maps, where each map location predicts bounding boxes and confidence scores, using a set of predefined default boxes for detecting objects of various sizes [10]. During training, these boxes are adjusted to achieve better alignment with the target defect [10].

Liu et. Al. demonstrated this algorithm enables real-time execution, providing both speed and accuracy when such requirements are critical, while maintaining lower computational complexity compared to Faster R-CNN [10]. Experimental results [53] demonstrated that the classical SSD model is effective for fabric defect detection, successfully identifying multiple defect categories such as holes, oil stains, surface debris, and broken yarns, confirming its suitability for textile inspection tasks. However, the baseline SSD often generated disproportionately large bounding boxes when detecting very small defects, leading to reduced detection confidence.

Some handcrafted variations, as in Liu et al. [54], created the Lightweight SSD (LW-SSD), a parallel dilated convolution method that utilizes dual-channel attention mechanisms. This approach achieves a 58% parameter reduction while maintaining a 48 fps detection speed, providing technical support for real-time detection on embedded platforms. Xie et al. [55] enhanced the traditional SSD by adjusting default boxes for long strip defect detection, demonstrating effective performance with high-speed detection and accurate detection of various defects on fabric surfaces with different textures in some datasets, achieving a mAP of 42.8% compared to Faster R-CNN (36.3%) and YOLOv3 (38.0%).

Authors of [55] introduced an enhanced SSD model for detecting fabric defects in real time. Their CNN-based system demonstrated high speed and robustness, particularly in identifying multiple defect types within complex textile structures. Nonetheless, the authors cited limitations including the need for large annotated datasets and performance sensitivity to fabric scale variations, which could lead to missed detections in fine-grained defects.

iv. YOLOv5:

YOLO family framework is characterized by its single-shot detection structure consisting of a backbone, neck and head components, using a single neural network process to predict bounding boxes and class probabilities [9], and it has been increasingly adopted, as it aligns well with industrial requirements, offering real-time object detection, high accuracy, computational efficiency, and suitability for deployment on edge devices [56].

By reducing network parameters using Cross-Stage Partial Network (CSPNet) in the head and PANet in the neck, along with residual structures and auto-anchor mechanisms, the model achieves faster and more lightweight performance compared to earlier YOLO versions [57]. It also introduced built-in Mosaic augmentation, which randomly combines four images with random sizes, allowing the detection of smaller objects accurately and reducing dataset size requirements [58].

The authors of [59] propose an improved YOLOv5s6 model integrating new advanced modules such as C2f (cross-stage partial bottleneck with two convolutions), BotTransformer and GeLU (Gaussian Error Linear Unit), achieving improvements of 11% to 12% in mAP0.5:0.95 compared with the standard YOLOv5s6 model. Their work obtains substantially better results than the yarn hairiness evaluation machine USTER TESTER 3, which they also use as a comparison.

The authors of [60] propose a YOLOv5-based algorithm combined with Large Selective Kernel (LSK) and an improved Lightweight Replication-based Generalized Feature Pyramid Network (Light-repGFPN), achieving a defect detection accuracy of 90.3% for flaw, dirt, fold, and hole types in nonwoven fabric. To obtain these results, a dedicated defect acquisition system was developed, comprising a stable light source, a hyperspectral camera positioned at a short distance, and a compact sample table. The analyzed defects exhibited high contrast against a uniformly white fabric background.

v. YOLOv8

YOLOv8 employs a backbone structure similar to YOLOv5 but replaces the CSPLayer with the newly introduced C2f module. The C2f enhances detection accuracy by integrating high-level features with contextual information [61]. YOLOv8 uses an anchor-free model with a decoupled head to process objectness, classification, and regression tasks independently. This design allows each branch to focus on its task and improves the model's overall accuracy [62].

The authors of [63] directly compared YOLOv5, YOLOv8 and Faster R-CNN for real-time fabric defect detection, and concluded that while YOLOv5 delivers higher detection speed at 118 FPS, its accuracy is comparatively lower, with a mAP of 0.686. In contrast, YOLOv8 attains a mAP of 0.756 with a speed of 103.5 fps, thereby surpassing both YOLOv5 and Faster R-CNN in terms of accuracy and efficiency.

Corroborating with that, ref. [56] presents a comparison between YOLOv8 and its predecessors, showcasing its superior performance, which achieves higher throughput with a comparable number of parameters, suggesting architecture-level optimizations that enhance hardware efficiency.

As demonstrated by official benchmarks [64], YOLOv5n achieves the fastest inference time for resource-limited equipment compared to other YOLO family frameworks, making

it a suitable choice for high-speed, hardware-optimized applications. Independent analysis, such as [65], also emphasizes the benefits of YOLOv5 in embedded systems, as it uses less power and utilizes less memory, while also demonstrating higher FPS (140 compared to 110 for YOLOv8) with lower mAP (0.503 compared to 0.568 for YOLOv8).

vi. YOLOv11:

YOLOv11 introduced key advancements by replacing the C2f block with the newly introduced C3k2 (Cross Stage Partial block with kernel size of 2), a more computationally efficient variant of the CSP bottleneck [66]. YOLOv11 also introduces the C2PSA (Convolutional block with Parallel Spatial Attention) block, which sets it apart from YOLOv8, as it lacks this specific attention mechanism. This mechanism strengthens spatial attention by guiding the model toward critical image regions, thereby improving detection accuracy for small or partially occluded objects [57].

A recent survey [57] states that the introduced enhancements improve performance in dense and complex textile patterns, enabling more accurate detection of overlapping and small-scale defects, and that by maintaining precise localization even under high-speed manufacturing conditions, these models strengthen the connection between research progress and industrial deployment, advancing real-time textile quality control.

The authors of [67] propose a variant of YOLOv11n, a lightweight, optimized version for edge deployment. The study does a comparison involving SSD, Faster R-CNN and 6 variants of the YOLO framework. It reports performance metrics such as mean Average Precision at IoU threshold 0.5 (mAP@0.5), frames per second, and Giga Floating Point Operations per Second (GFLOPS). Authors conclude that YOLOv11n is the most optimized model, consuming fewer computational resources and having the smallest model size. The proposed model outperforms YOLOv11n with a 2.1% increase in mAP@0.5.

A similar approach is carried out by [68] in which authors compare Faster R-CNN, Cascade R-CNN, AutoAssign, RT-DETRv2s, 6 YOLO variants and their custom variant. Results show the proposed variant outperformed all other models, achieving 0.693 and 0.707 mAP@0.5 on two datasets, respectively, while the YOLOv11n baseline achieved the second-highest scores.

vii. Comparison

Based on the state-of-the-art review, traditional algorithms remain prevalent due to their long-standing implementation and reliability. However, there is a notable and ongoing shift towards YOLO-based algorithms and their variants, driven by their architectural advantages, which support continuous performance improvements and facilitate integration into more comprehensive solutions.

A comparative framework was developed to highlight the most impactful features for the selection of the model to be further investigated. The comparison relies on evaluations reported by several authors [54,57,69,70] and considers key aspects relevant to visual fabric defect detection, namely accuracy, computational requirements, raw fabric detection, patterned fabric detection, and FPS. This comparative study resulted in the quantitative data presented in Table 1.

Following this study, accuracy was initially evaluated. The models demonstrating the highest accuracy were Faster R-CNN, YOLOv8n and YOLOv11n, a critical consideration for real-time applications in textile production lines. Regarding computational requirements, the YOLO models exhibited the lowest demands, making them particularly suitable for deployment in edge devices. In raw fabric defect detection, YOLOv11n and Faster R-CNN achieved superior performance, despite other models also delivering competitive results. For patterned fabric defect detection, YOLOv8n, YOLOv11n, and Faster R-CNN showed the most consistent performance. Finally, in terms of FPS, the YOLO-based architecture

achieved the highest rates, a feature that, combined with accuracy, is essential for real-time defect detection in production environments.

Table 1. Visual object detection models comparison (based on [54,57,69,70]).

Model	Accuracy	Computational Requirements	Raw Fabric Detection	Pattern Fabric Detection	FPS	Relevance
SSD	Medium/High	Low/Moderate	Good	Low	High	Useful speed vs. accuracy for raw fabric
Faster R-CNN	Very High	High	High	High	Low	High precision benchmark if latency is allowed
YOLOv5n	Medium/High	Very Low	Good	Good	Very High	Lightweight YOLO baseline, for low latency and very constrained environments
YOLOv8n	High	Very Low	Good	High	Very High	State-of-the-art efficiency for nano models.
YOLOv11n	High	Very Low	High	High	Very High	Good balance of accuracy and edge performance

EfficientDet was also included in the analysis, as it achieves competitive accuracy levels while maintaining low computational requirements, which favors its implementation on edge devices [71]. However, the limited number of comparative studies and references in the literature prevents conclusive assessments of its capabilities relative to the other models considered.

From the comparative analysis, YOLOv11n was identified as the most promising model. As one of the most recent versions of the YOLO family, it combines a lightweight architecture suitable for deployment on edge devices with real-time detection capabilities and high accuracy in fabric defect detection. Although Faster R-CNN also achieved competitive results, its high computational cost renders it impractical for edge-based applications. Consequently, YOLOv11n is considered the most suitable candidate for further experimentation in the subsequent stages of this study.

3. Methodology

This section describes the methodology employed for the development and deployment of an AI-based computer vision system for fabric defect detection in a real industrial environment. Among the various types of fabrics produced, solid-color uniform cotton and linen fabrics with weights between 120 and 160 g/m^2 are the most prevalent and are manufactured in continuous production. Consequently, the system was primarily designed for the detection of defects in solid-color uniform fabrics. Nevertheless, it is necessary to maintain versatility across various color variations (e.g., white, green, brown, navy blue, black) and fabric weights, which impact material transparency. Darker colored textiles are particularly challenging for color bleeding and crease detection due to their low contrast. Additionally, lower thickness introduces noise, as the fabric material has see-through capabilities.

Figure 2a presents an overview of the existing loom equipment in the factory, consisting of multiple mechanical rollers that regulate and maintain uniform tension across the fabric, ensuring it remains evenly stretched during the unrolling process. The image clearly

shows the transition between two different fabrics, one lighter and the other darker in tone. On the other hand, Figure 2b shows fabrics with different weights, highlighting the issue related to the material's transparency.



(**a**) (**b**)

Figure 2. Example of rolling loom equipment in the factory: (**a**) side-view showing various rollers responsible for stretching the fabric; (**b**) transition between different fabric samples.

The objective is to develop a solution capable of detecting anomalies, like holes, creases and color bleeding, in a rolling fabric with 1200 m of length in real time at an average speed of 15 m/min, with high accuracy and efficiency, while minimizing latency and dependence on manual inspection.

This section is organized into three main components:

- System architecture: the end-to-end pipeline design, including hardware and software components;
- Deployment setup: the implementation and integration of the system in a production environment;
- Performance evaluation metrics: the criteria used to assess the system's performance.

By addressing the functional and environmental requirements of the deployment scenario, the methodology ensures that the solution is robust, scalable, and aligned with the operational needs of textile production lines.

3.1. System Architecture

The proposed system extends the conventional architecture for computer vision-based defect detection, as illustrated in Figure 3, by integrating edge AI deployment, defect localization, and cloud data storage. These enhancements are specifically designed to address the real-time requirements of fabric inspection in industrial environments.

The architecture includes the following core components:

- Model Training: A labeled dataset of fabric images, including three defect types with one being predominant, the crease defect, as shown in Table 2, is used to train a deep learning model on a Graphics Processing Unit (GPU) cluster. The trained model is optimized (e.g., pruned, quantized) for edge deployment.
- Image Acquisition: RGB-D cameras (Intel RealSense D435i) installed on the production line continuously capture image frames of the rolling fabric.

- Edge Device (NVIDIA Jetson Orin Nano): This device executes the AI inference pipeline locally, minimizing latency and enabling real-time defect detection without relying on constant cloud communication.
- Visual Feedback: The output of the detection process is presented in real time through a monitoring dashboard, by distance through RTMP Streaming or locally saved recordings. This allows operators to supervise and analyze the inspection process.
- Database: Detected defects are stored on a cloud database for traceability, reporting, and further statistical analysis.

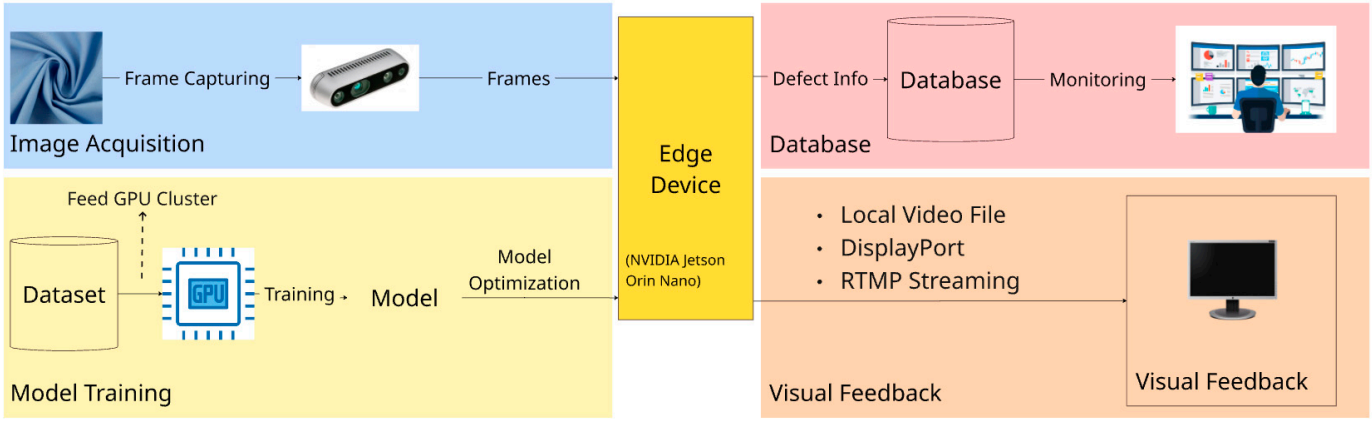


Figure 3. Proposed system architecture for the implemented computer vision-based defect detection.

Table 2. Distribution of the number of instances per defect in the dataset.

Defect Type	Instances
Hole	2169
Crease	27,126
Color bleeding	1365
Total	30,660

The imbalance present in this dataset, as shown in Table 2, is caused by the fact that creases are the most common defect in the company's use case, so they appear in a more substantial number per frame, even in images showing other types of defects.

This modular architecture supports scalability and real-time operation while ensuring seamless integration with existing industrial infrastructure. A more detailed analysis of the constitution and functionalities in this architecture is carried out in the following topics.

i. Image acquisition

Among the available imaging alternatives, the Intel RealSense D435i camera was selected due to its cost-effectiveness and availability within the laboratory environment. This device provides RGB images at a resolution of 1920 × 1080 pixels and depth images at 1280 × 720 pixels, supporting frame rates of up to 90 FPS [72]. Its compact design and affordability render it a suitable choice for experimental validation. Preliminary evaluations indicate that the camera performs reliably under controlled lighting conditions, exhibiting satisfactory accuracy in the detection of structural defects such as creases and perforations. However, its effectiveness is reduced when identifying chromatic anomalies, including discoloration and subtle color variations.

The looms employed by the textile manufacturer have a working width of 3.4 m, while the fabrics processed typically do not exceed 3 m in width. To ensure complete coverage of the inspection area and to minimize the risk of missing small-scale defects, two cameras were deployed. These cameras were used not only to capture raw image data but also to

support subsequent inference tasks. Due to the field of view of each camera, an overlap region is present, allowing the same defect to be detected by both devices, which enhances detection reliability.

ii. Pre-processing

In this study, images were initially acquired at a resolution of 1920 × 1080 pixels to ensure high detail. To maintain compatibility with the YOLO architecture and preserve model performance, all images were subsequently resized to the standard input size of 640 × 640 pixels. Grayscale conversion was considered as a potential preprocessing step to enhance the visibility of structural defects such as holes and creases; however, it was ultimately discarded, as it significantly impairs the detection of chromatic anomalies, such as color bleeding, which relies on color information.

To address real-world industrial conditions—including illumination variability and noise induced by vibrations from legacy loom systems—a series of data augmentation techniques was applied. These included controlled variations in brightness within the range of −25 % to +25 % and the addition of Gaussian noise up to 0.5 % of the pixels. To further improve the model's robustness and generalization to diverse defect types, additional augmentation was employed, including random rotations within a range of −15 ° to +15° (degrees). Worth mentioning, YOLO's built-in mosaic augment flag parameter was evaluated, and performance was compared with and without this option.

All these techniques were evaluated incrementally to assess their individual impact on model performance. Results are presented in Table 3. Analyzing the obtained mAP@50 values, the impact of augmentation techniques on improving detection performance is evident. Notably, brightness adjustment consistently contributes to enhanced results. Although individual augmentation techniques yield only marginal improvements, the combination of all techniques leads to a substantial increase in overall defect detection effectiveness, achieving a mAP@50 of 0.821.

Table 3. Comparison of different augmentation techniques.

Augmentation Techniques	mAP@50
No Augmentation	0.751
Mosaic	0.806
Mosaic + Brightness	0.802
Mosaic + Noise	0.781
Mosaic + Rotation	0.804
Mosaic + Brightness + Noise	0.811
Mosaic + Brightness + Rotation	0.793
Mosaic + Noise + Rotation	0.767
Mosaic + Brightness + Noise + Rotation	0.821

iii. Feature Extraction

YOLOv11's integrated feature extraction modules were employed to generate feature maps encoding information related to textures, shapes, and object components. Several of these modules represent advancements over previous YOLO versions, including the C3k2 block and the component. The C3k2 block is applied in the early stages of the network to perform downsampling by progressively reducing spatial resolution while increasing the number of channels, thereby enhancing computational efficiency. The C2SPA module implements a spatial attention mechanism that enables the model to prioritize salient regions within the image, thereby improving detection accuracy for defects of varying sizes and positions.

iv. Classification

Classification is performed in the model's head, which functions as the prediction mechanism by providing object localization and classification based on the extracted feature maps. This component consists of a series of Conv2D layers that reduce the feature representations to the required outputs for bounding box coordinates and class predictions, which the final Detect Layer subsequently processes.

3.2. Setup Implementation

The proposed defect detection system was deployed on a real industrial textile inspection line.

The implementation required careful integration of the hardware components with the physical environment and adaptation to operational constraints, as shown in Figure 4. The conveyor system rolled the fabric at a constant speed of approximately 15 m per minute, similar to real production conditions.

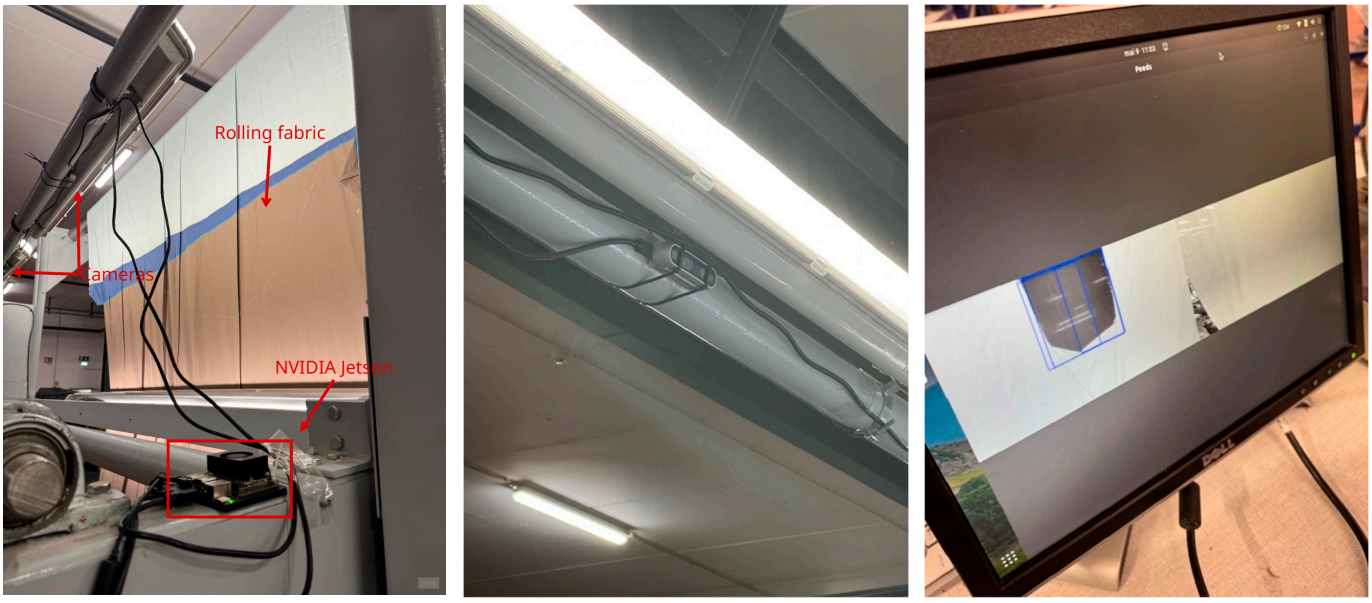


Figure 4. Real industrial environment setup of the fabric defect detection system: (**a**) general overview; (**b**) detailed integration of the camera with the lighting; (**c**) monitoring visualization feedback for the user with emphasis on defect recognition.

Figure 4a presents a general overview of the existing textile mechanical system in the factory, showing the integration of the proposed system. To ensure consistent illumination, lamps were strategically integrated in proximity to the cameras, as shown in Figure 4b.

The integration of this system must follow the add-on concept, being easy to integrate with existing systems, which may have differences between them, from manufacturer to manufacturer, but should not be invasive. In other words, it should not require mechanical changes to existing systems in the factory. Thus, the installation process required minor mechanical adjustments to accommodate the hardware and ensure safety, durability, and stability under continuous operation conditions.

The setup was implemented on a fabric rolling line, with the following key aspects:

- Camera positioning: As shown in Figure 4a, two Realsense D435i RGB-D cameras were installed above the rolling fabric, fastened securely to a metallic pipe at a fixed distance of approximately 1 m from the fabric. This configuration enabled a field of view of approximately 1 m in height and 3 m in width. The cameras were oriented

perpendicularly to the fabric surface, aligned with the direction of fabric movement, and positioned close to the light sources to reduce shadows and reflections. Each camera was connected via Video for Linux Two (V4L2) and processed locally on the Jetson Orin Nano. The camera feeds were configured at a resolution of 640 × 480 pixels and a frame rate of 30 FPS.

- Edge Device Integration: An NVIDIA Jetson Orin Nano, NVIDIA, Santa Clara, CA, USA, device was securely mounted near the cameras, with all cables routed and fixed to avoid interference with machine operation. This close proximity minimizes data transmission latency and supports real-time inference.
- Lighting Adjustments: 120 cm width supplemental diffuse LED lights were installed above the fabric to ensure consistent illumination across the surface, as shown in Figure 4b. Poor lighting conditions from the industrial floor initially caused false positives due to shadows and inconsistent texture appearance. By adjusting the angle and intensity of the lighting, these issues were mitigated.
- A display was connected to the NVIDIA Jetson Orin Nano edge device using DisplayPort, showing the real-time camera feed with bounding boxes over detected defects. This visual feedback interface provides operators with immediate insight into the inspection process, as shown in Figure 4c. The system continuously analyzes the incoming frames and overlays the localization results, which are then recorded and stored.
- Defect metadata and localization info are sent via MQTT to a central database for traceability, remote access and historical analysis.

3.3. Dataset Preparation and Model Training

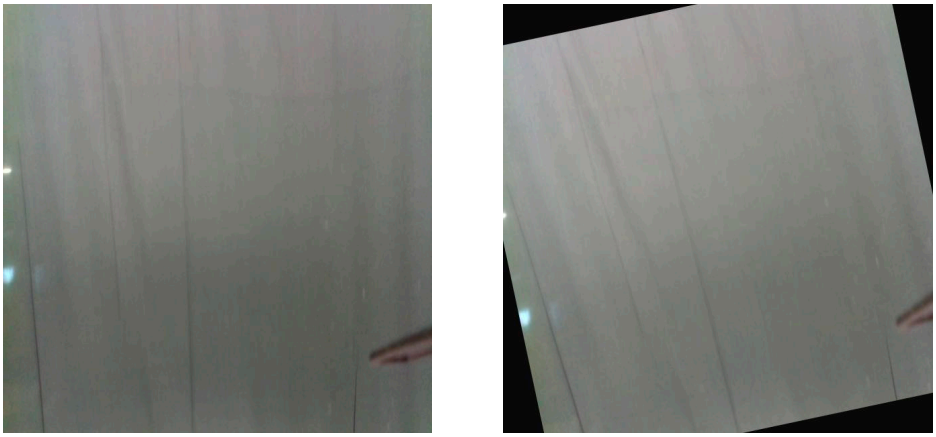
A custom dataset was built collecting video directly from the production line using the same camera model and setup described previously, in a setup similar to Figure 4. The working loom was video recorded to replicate the real inspection conditions. The recordings were processed using an algorithm that extracted one frame every 30 frames, resulting in one image per second, for each camera. This procedure resulted in a dataset comprising 5104 annotated fabric images. Due to the large volume of collected images, only a subset was annotated. Image annotation was performed using the Roboflow platform (https://roboflow.com/), adhering to the YOLO format. Three defect classes were defined: Hole, Crease, and Color Bleeding. To simulate the variability typically encountered in industrial environments, data augmentation techniques were applied, resulting in a significant increase in dataset size to 12,774 images. These augmentations aimed to improve the generalization capability of the detection model by introducing variations in lighting, orientation, scale, and other visual attributes.

An example of data augmentation is illustrated in Figure 5, where image rotation is applied. This technique increases the diversity of visual patterns by altering the orientation of defects, thereby enhancing the robustness and generalization capability of the detection algorithm across different fabric alignments and defect configurations.

Another example of data augmentation is brightness adjustment, as illustrated in Figure 6. This approach generates variations in image contrast for the same defect type, enabling the model to learn more effectively under differing lighting conditions. As a result, the system becomes more resilient to illumination changes commonly encountered in industrial environments.

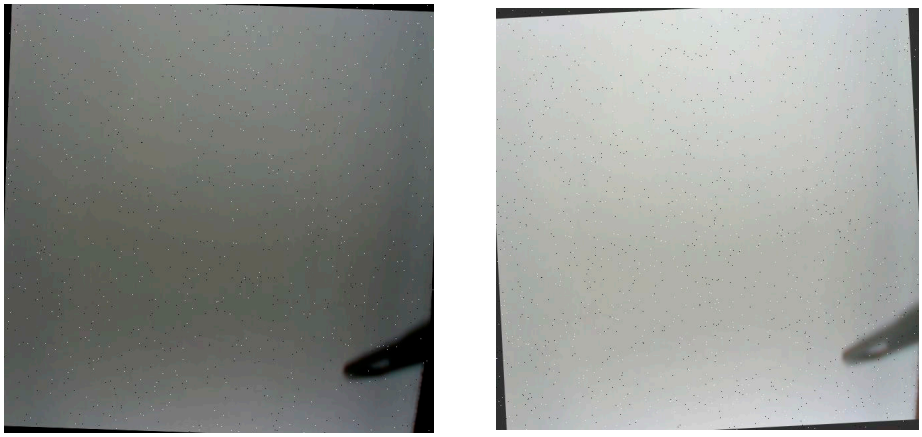
Another form of data augmentation involves the addition of noise, as shown in Figure 7. Introducing noise into the training images helps prevent the model from overfitting to overly specific visual features of the defects. This strategy enhances the model's

ability to generalize, enabling it to detect defects reliably even in the presence of signal interference or image degradation.



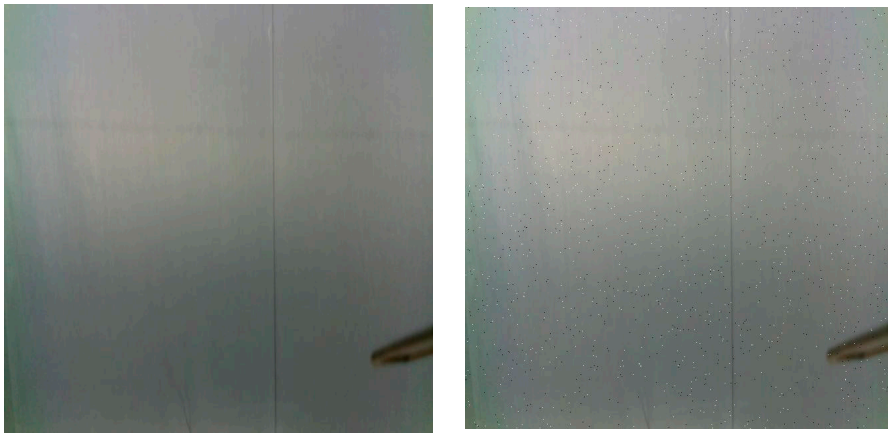
(**a**) (**b**)

Figure 5. Example of rotation augmentation: (**a**) original image; (**b**) rotated image.



(**a**) (**b**)

Figure 6. Example of brightness adjustment augmentation: (**a**) image with lower brightness; (**b**) image with higher brightness.



(**a**) (**b**)

Figure 7. Example of noise addition augmentation: (**a**) original image; (**b**) image with added noise.

The dataset was divided into training (75%), validation (15%) and test (10%) subsets. Model training was carried out using YOLOv11n, a lightweight variant optimized for edge

deployment scenarios. The training process employed a batch size of 16 over 1000 epochs, with early stopping after 50 epochs with no improvements based on the mAP computed on the validation set. Upon completion, the model was validated and subsequently exported to the TensorRT (.engine) format, enabling optimized inference performance on NVIDIA Jetson platforms.

3.4. Real-Time Processing System

Given the stringent requirements for real-time defect detection and data privacy, cloud-based processing was considered unsuitable due to potential latency, bandwidth constraints, and increased risk of data loss. In time-sensitive applications such as textile inspection, even minor delays in detection can result in unnoticed defects, leading to reduced product quality and increased material waste.

To address these challenges, local (edge) processing was adopted, offering low-latency inference, independence from network connectivity, and enhanced system reliability. The selected hardware platform was the NVIDIA Jetson Orin Nano 8 GB, a compact and efficient edge AI module specifically designed for embedded AI workloads.

As illustrated in Figure 8, the NVIDIA Jetson Orin Nano is equipped with a high-performance GPU based on the NVIDIA Ampere architecture, featuring 1024 CUDA cores and 32 Tensor cores. This configuration enables efficient and high-speed deep learning inference directly at the edge, close to the point of data acquisition, thereby minimizing latency and supporting real-time processing requirements.



Figure 8. NVIDIA Jetson Orin Nano [73].

The edge device was tasked with the following operations:

- Capturing image streams from the RGB-D cameras;
- Executing the inference pipeline using an optimized deep learning model;
- Displaying real-time detection results on a local monitor;
- Transmitting detection metadata to the database for logging and traceability purposes.

Owing to its hardware acceleration capabilities and efficient power consumption (below 15 W), the Jetson Orin Nano delivered sufficient processing throughput without the need for external cooling systems or extensive infrastructure.

Inference performance was evaluated directly on the Jetson Orin Nano using a model optimized and converted to TensorRT with FP16 precision. The corresponding results are presented in Table 4.

Table 4. Inference speed benchmark.

Metric	Value
Inference Time (ms)	~10 ms
FPS	~100 fps

The achieved throughput of approximately 100 FPS ensures that the system comfortably meets real-time processing requirements, even under high fabric speeds and multi-camera input scenarios. This performance margin provides scalability for future enhancements, such as higher-resolution image inputs or the inclusion of additional defect classes, without necessitating significant hardware modifications.

Furthermore, leveraging an edge device confers advantages including improved data privacy, reduced dependence on network connectivity, and immediate feedback, thereby constituting a robust and production-ready solution for continuous industrial deployment.

3.5. Evaluation Metrics

To objectively assess the performance of the fabric defect detection system, both detection quality and inference efficiency were evaluated using standard computer vision metrics. The evaluation focused on the system's capability to accurately identify and localize fabric defects. The methodology involved the following metrics.

The first metric considered was the precision (P), which quantifies the proportion of correctly identified positive detections relative to all positive predictions. Precision is defined considering Equation (1), where TP represents the number of true positives and FP the number of false positives.

$$P = \frac{TP}{TP + FP} \tag{1}$$

The second metric was recall (R), which measures the proportion of actual defects correctly detected by the system, thus reflecting the sensitivity of the model. Recall is given by Equation (2), where FN denotes false negatives. High recall indicates a low rate of missed defects.

$$R = \frac{TP}{TP + FN} \tag{2}$$

Once P and R are obtained, the $F1$ factor can be calculated, with the help of Equation (3), as the harmonic mean of these two metrics, providing a balanced measure that accounts for both false positives and false negatives.

$$F1 = \frac{2 \times P \times R}{P + R} \tag{3}$$

The next parameter was the average precision (AP). To determine the AP, Equation (4) was used, integrating the precision-recall curve. AP represents the area under this curve for a given defect class, summarizing the trade-off between Precision and Recall across different thresholds.

$$AP = \int_0^1 P(R) dR \tag{4}$$

To extend this to multi-class scenarios, the mAP was used. As shown in Equation (5), mAP is the average of AP values across all C classes. Model performance was evaluated using mean Average Precision at an IoU threshold of 0.5 (mAP@0.5). Here, IoU is defined as the ratio of the area of overlap to the area of union between the predicted bounding box and the ground-truth bounding box. It is particularly relevant for evaluating the overall performance of multi-class object detection systems.

$$mAP = \frac{\sum_{c=1}^{C} AP(c)}{C} \tag{5}$$

Given the real-time nature of the application, latency and throughput were also critical evaluation factors. The inference speed was measured as the time taken by the model to process a single image on the edge device, as expressed in Equation (6).

$$FPS = \frac{1000}{Inference\ Speed(ms)} \tag{6}$$

Together, these metrics provide a comprehensive framework for evaluating the performance of the fabric defect detection system. Precision, Recall, and F1 score provide insights into the model's capability to correctly identify defects while balancing false positives and false negatives. *AP* and *mAP* offer a more granular assessment across multiple defect classes and decision thresholds, which is essential for multi-class classification tasks. Additionally, inference speed, expressed in *FPS*, ensures that the model complies with the real-time processing requirements characteristic of industrial fabric inspection. The combined use of these quantitative indicators validates the system's effectiveness both in terms of detection accuracy and operational reliability under production conditions.

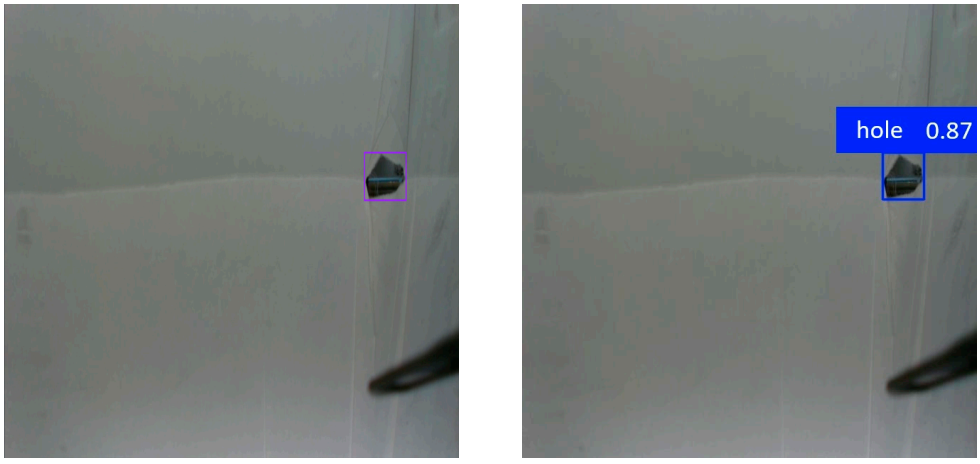
4. Experimental Results

This section presents the different experimental results obtained, before and after processing, for the different fabric defects: (i) hole; (ii) color bleeding; (iii) crease.

Following visual detection examples for each defect type, a performance metrics table summarizes the model's Precision, Recall, F1 score, and mean Average Precision (mAP@50) across classes.

4.1. Hole Defect Detection

Figure 9a illustrates an example of a hole located at the right edge of the fabric image. Figure 9b shows the detection system correctly identifying this defect as a "hole" with a confidence score of 0.87, indicated by the blue bounding box. The confidence level is relatively high, which demonstrates the model's sensitivity to this type of defect, as well as its capability to effectively differentiate such features from the background.



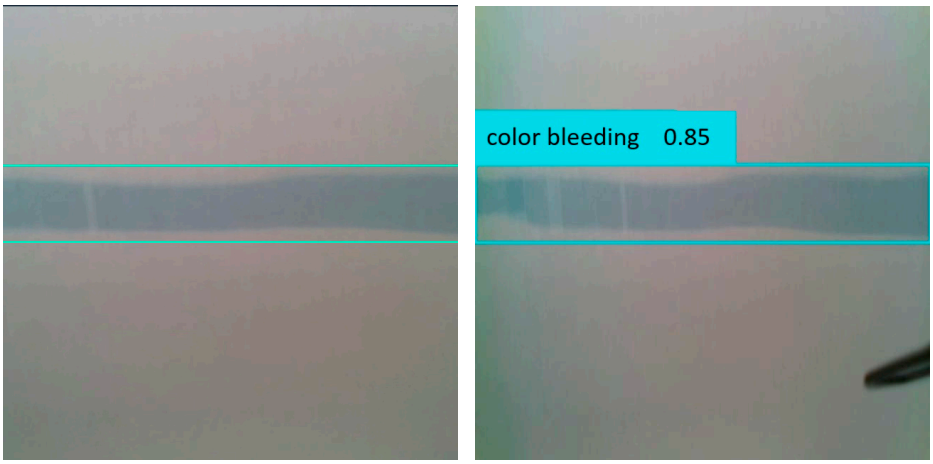
(**a**) (**b**)

Figure 9. Example of hole detection: (**a**) manual user identification; (**b**) image processing identification.

4.2. Color Bleeding Detection

The second case, shown in Figure 10, depicts a faded area located at the bottom of the image. The original image shown in Figure 10a reveals a horizontal band where the fabric exhibits reduced saturation, likely caused by wear or printing inconsistencies. In

Figure 10b, the model accurately identifies this region as color bleeding, assigning a high confidence score of 0.85, which reflects strong certainty in the classification.

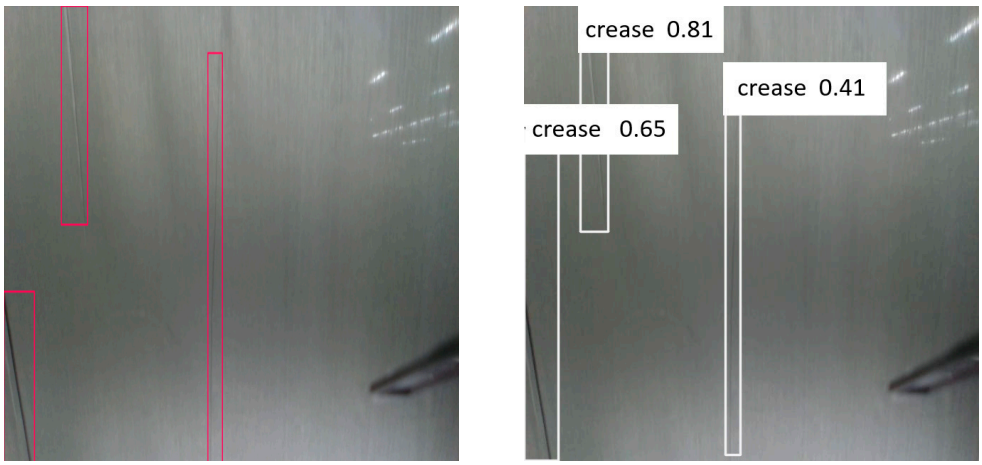


(**a**) (**b**)

Figure 10. Example of color bleeding detection: (**a**) manual user identification; (**b**) image with processing.

4.3. Crease Detection

The third example, presented in Figure 11, illustrates vertical creases across the fabric surface. In the annotated image shown in Figure 11a, multiple vertical lines are evident, suggesting defects caused by folding or tension irregularities. Figure 11b demonstrates that the model successfully detects these features as creases, assigning confidence scores that vary across the detections, with the highest score reaching 0.81. Lower-confidence predictions, such as 0.41 and 0.65, are also observed, reflecting uncertainty in certain areas, which may correspond to noise or faint crease patterns.



(**a**) (**b**)

Figure 11. Example of crease detection: (**a**) manual user identification; (**b**) image with processing.

4.4. Performance Metrics

Following visual validation, the overall performance of the system was quantitatively assessed using the object detection metrics introduced in Section 3.5 The evaluation results are summarized in Table 5, presenting detailed performance metrics for each defect class individually, as well as the overall average across all classes.

The model demonstrates solid overall performance, achieving a precision of 0.75, a recall of 0.78, an F1 score of 0.765, and a mAP@50 of 0.821. These results indicate a well-balanced trade-off between detection accuracy and completeness.

Table 5. Model performance evaluation metrics.

Classes	Precision	Recall	F1 Score	mAP(@50)
all	0.750	0.780	0.765	0.821
hole	0.890	0.760	0.820	0.870
color bleeding	0.620	0.810	0.702	0.850
crease	0.730	0.760	0.745	0.750

Analyzing individual classes, the hole class exhibits the highest performance, with a precision of 0.89 and a mAP of 0.87, although its recall of 0.76 suggests that some instances were not detected. The color bleeding class achieved the highest recall (0.81), indicating strong sensitivity to true positives; however, its lower precision (0.62) points to a higher rate of false positives. The crease class presents balanced performance across metrics, with an F1 score of 0.745 and the lowest mAP (0.75), highlighting opportunities for improvement in localization accuracy.

Overall, the model proves to be reliable and capable of real-time operation. Key areas for further enhancement include reducing false positives in the color bleeding class and improving recall for the hole class.

5. Discussion

Following the implementation of the proposed methodology and its integration into the NVIDIA Jetson Orin Nano platform, an experimental evaluation was conducted. This process focused on analyzing multicolor, patternless cotton and linen fabric, as it is the textile company's most frequently manufactured product. The system aimed to detect three common defect types: (i) holes, (ii) color bleeding, and (iii) creases. Given the range of defects targeted, certain limitations in system performance were anticipated, particularly due to incompatibilities in applying uniform preprocessing strategies across different defect classes. This challenge highlights the need for hybrid solutions tailored to specific defect characteristics, thereby optimizing overall system performance. Nevertheless, the primary objective was to develop a low-cost, compact, and versatile system compatible with a variety of loom configurations. Based on the experimental data, the system achieved a mean Average Precision (mAP) of over 75%, indicating a consistent level of robustness in detecting the specified defects. Although the overall performance can be considered modest, it is adequate for the intended application within a broader, multi-stage quality control pipeline. The per-class accuracy scores were 0.89 for holes, 0.62 for color bleeding, and 0.73 for creases. Analysis of the results indicates that the system performs more effectively in detecting physical anomalies (such as holes and creases) than visual defects like color bleeding. This disparity is primarily attributed to two factors: the characteristics of the imaging hardware (as discussed in Section 2.1) and the absence of defect-specific preprocessing techniques:

- The camera employed is optimized for capturing depth and structural features, enhancing the system's ability to detect physical deformations. However, its limited color sensitivity impairs performance in identifying chromatic irregularities.
- Additionally, no preprocessing techniques tailored to specific defect types were employed. While methods such as contrast enhancement or grayscale adjustment could improve the detection of particular features, they were intentionally omitted to preserve the generalizability of the system. The detection model was designed to operate without class-specific preprocessing, aiming to provide unified detection capabilities across both physical and visual defects.

Alternative strategies to enhance the detection of color-related defects, such as color bleeding, could involve the application of sensor fusion using multispectral cameras. These cameras can isolate the defect's color wavelength and highlight the defect. The use of these cameras could also be useful for detecting non-chromatic defects, such as holes, by using a distinctive colored background panel. Another alternative could be the manipulation of the image in the HSV color space by adjusting the Hue, Saturation, and Value attributes of the image, highlighting the defect. This approach, although very useful for chromatic defect detection, requires prior knowledge of the current fabric HSV values and constant updating as different color fabric loops are placed throughout the loom and also could negatively affect the detection of other defects such as creases.

Considering the imbalance in the defects present in the dataset, the model is prone to being biased towards the majority defect, having poor generalization for the minority ones. Another concern is that it can have misleading performance metrics, for example, having 95% accuracy in the majority class and 5% in minorities. One solution could be rebalancing by eliminating a proportion of these images but even in frames with other defects, creases are still present. Given that creases are present even in images with other types of defects, augmentation exacerbates the imbalance even further.

This study focused on detecting defects in plain, solid-colored cotton and linen fabrics, representing a specific industrial scenario. However, this choice introduces a notable limitation in the model's generalizability, as it may not perform reliably on other common fabric types, such as patterned, textured, or multicolored materials.

6. Conclusions and Further Work

This study presented the implementation and integration of a defect detection system tailored for the textile industry. The proposed solution is cost-effective, compact, and easily adaptable to various types of textile looms. It demonstrated versatility in detecting common fabric defects such as holes, color bleeding, and creases. Experimental results confirm the system's balanced and consistent performance across the targeted defect classes, while its small physical footprint facilitates straightforward integration into constrained industrial environments.

For future work, it would be beneficial to enhance the algorithms by incorporating preprocessing steps tailored to each type of defect, in order to improve the individual identification of each defect. It would also be of interest to evaluate the system's performance under different background conditions, such as constant backlighting or a uniform color background, to enhance contrast. Another relevant direction for improving the individual identification of defects is the use of sensor fusion, combining different types of cameras. While the current setup has shown promising results in detecting holes and creases, additional cameras—better suited for identifying issues such as color bleeding—could complement the existing system by integrating data from multiple sources.

Another promising direction involves assessing the system's performance over extended operational periods, such as an entire workday. This includes analyzing its behavior over the full length of fabric rolls and under mechanical vibrations typical of loom operation. Based on these insights, the development of specialized mechanical mounts or dampers for the cameras may improve system stability and consistency across different loom configurations.

Overdetection, caused by common areas captured by both cameras, can also happen, and it is a needed optimization in order to perform an accurate evaluation in real-world settings.

Given the dataset's limitations—specifically non-textured and non-patterned fabrics—future work should aim to expand the dataset to encompass a broader diversity of fabric

types. This can be achieved through the collection of images in real-world environments or the generation of synthetic data. Such an approach is expected to enhance model robustness and improve its applicability across a wider range of real-world textile scenarios.

Author Contributions: Conceptualization, R.M., F.D.d.S. and H.C.; methodology, R.M., L.A.M.B. and V.V.; software, R.M.; validation, L.A.M.B., F.D.d.S., V.C. and H.C.; formal analysis, R.M. and V.V.; investigation, R.M. and V.V.; data curation, R.M.; writing—original draft preparation, R.M. and L.A.M.B.; writing—review and editing, all authors; visualization, R.M., L.A.M.B. and V.V.; supervision, H.C. and V.C. All authors have read and agreed to the published version of the manuscript.

Funding: Vítor Carvalho cooperation was funded by FCT/MCTES grant numbers UIDB/05549:2Ai and UIDP/05549:2Ai.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Acknowledgments: The images used to validate the work performed in this paper were provided by Textêis J.F.Almeida, S.A.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

2D	Two-Dimensional
3D	Three-Dimensional
AI	Artificial Intelligence
AOI	Automated Optical Inspection
AP	Average Precision
C2f	Cross-stage partial bottleneck with two convolutions
C2PSA	Convolutional block with Parallel Spatial Attention
C3k2	Cross Stage Partial with kernel size 2
CNN	Convolutional Neural Network
CSPNet	Cross-Stage Partial Network
CSV	Comma-Separated Values
CUDA	Compute Unified Device Architecture
DCN	Deformable Convolution Network
FN	False Negatives
FP	False Positives
FP16	Floating-Point 16bits
FPS	Frames Per Second
GA	Genetic Algorithm
GB	Giga Bytes
GLCM	Gray-Level Co-occurrence Matrix
GPU	Graphics Processing Unit
HOG	Histogram of Oriented Gradients
IoU	Intersection over Union
K-NN	K-Nearest Neighbours
LED	Light-Emitting Diode
Light-repGFPN	Lightweight Replication-based Generalized Feature Pyramid Network
LSK	Large Selective Kernel
LW-SSD	Single Shot Detector
mAP	Mean Average Precision
MES/ERP	Execution System/Enterprise Resource
ML	Machine Learning
mm	millimeter

MPCA	MaxPool with Coordinate Attention
MQTT	Message Queuing Telemetry Transport
ms	milliseconds
P	Precision
R	Recall
ReLU	Rectified Linear Unit
RGB	Red, Green and Blue
RGB-D	Red, Green, Blue and Depth
RTMP	Real-Time Messaging Protocol
SSD	Single Shot Detector
SVM	Support Vector Machine
TP	True Positives
YOLO	You Only Look Once

References

1. Associação Têxtil E de Vestuário. Available online: https://atp.pt/pt-pt/estatisticas/caraterizacao/ (accessed on 10 July 2025).
2. Circularity of the EU Textiles Value Chain in Numbers. Available online: https://www.eea.europa.eu/en/analysis/publications/circularity-of-the-eu-textiles-value-chain-in-numbers (accessed on 9 July 2025).
3. Kang, X.; Zhang, E. A Universal Defect Detection Approach for Various Types of Fabrics Based on the Elo-Rating Algorithm of the Integral Image. *Text. Res. J.* **2019**, *89*, 4766–4793. [CrossRef]
4. Carrilho, R.; Yaghoubi, E.; Lindo, J.; Hambarde, K.; Proença, H. Toward Automated Fabric Defect Detection: A Survey of Recent Computer Vision Approaches. *Electronics* **2024**, *13*, 3728. [CrossRef]
5. Li, M.; Wan, S.; Deng, Z.; Wang, Y. Fabric Defect Detection Based on Saliency Histogram Features. *Comput. Intell.* **2019**, *35*, 517–534. [CrossRef]
6. Li, F.; Yuan, L.; Zhang, K.; Li, W. A Defect Detection Method for Unpatterned Fabric Based on Multidirectional Binary Patterns and the Gray-Level Co-Occurrence Matrix. *Text. Res. J.* **2020**, *90*, 776–796. [CrossRef]
7. Khwakhali, U.S.; Tra, N.T.; Tin, H.V.; Khai, T.D.; Tin, C.Q.; Hoe, L.I. Fabric Defect Detection Using Gray Level Co-Occurence Matrix and Local Binary Pattern. In Proceedings of the 2022 RIVF International Conference on Computing and Communication Technologies (RIVF), Ho Chi Minh City, Vietnam, 20–22 December 2022; pp. 226–231.
8. Lizarraga-Morales, R.A.; Correa-Tome, F.E.; Sanchez-Yanez, R.E.; Cepeda-Negrete, J. On the Use of Binary Features in a Rule-Based Approach for Defect Detection on Patterned Textiles. *IEEE Access* **2019**, *7*, 18042–18049. [CrossRef]
9. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 779–788.
10. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In *Computer Vision—ECCV 2016, Proceedings of the 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016*; Leibe, B., Matas, J., Sebe, N., Welling, M., Eds.; Springer International Publishing: Cham, Switzerland, 2016; pp. 21–37.
11. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [CrossRef]
12. Taiwo, O.; Ezugwu, A.E.; Oyelade, O.N.; Almutairi, M.S. Enhanced Intelligent Smart Home Control and Security System Based on Deep Learning Model. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 9307961. [CrossRef]
13. Chowdhury, M.E.H.; Rahman, T.; Khandakar, A.; Ayari, M.A.; Khan, A.U.; Khan, M.S.; Al-Emadi, N.; Reaz, M.B.I.; Islam, M.T.; Ali, S.H.M. Automatic and Reliable Leaf Disease Detection Using Deep Learning Techniques. *AgriEngineering* **2021**, *3*, 294–312. [CrossRef]
14. Munappy, A.R.; Bosch, J.; Olsson, H.H.; Arpteg, A.; Brinne, B. Data Management for Production Quality Deep Learning Models: Challenges and Solutions. *J. Syst. Softw.* **2022**, *191*, 111359. [CrossRef]
15. Shorten, C.; Khoshgoftaar, T.M. A Survey on Image Data Augmentation for Deep Learning. *J. Big Data* **2019**, *6*, 1–48. [CrossRef]
16. Bai, J.; Wu, D.; Shelley, T.; Schubel, P.; Twine, D.; Russell, J.; Zheng, X.; Zhang, J. A Comprehensive Survey on Machine Learning Driven Material Defect Detection. *ACM Comput. Surv.* **2025**, *57*, 1–36. [CrossRef]
17. Kumar, A. Computer-Vision-Based Fabric Defect Detection: A Survey. *IEEE Trans. Ind. Electron.* **2008**, *55*, 348–363. [CrossRef]
18. Nacy, S.M.; Abbood, W.T. Automated Surface Defect Detection Using Area Scan Camera. *Innov. Syst. Des. Eng.* **2013**, *4*, 1–10.
19. Chen, Y.L.; Tsai, C.W.; Ding, F.S.; Hsu, Q.C. Application of Line Scan Technology for Defect Inspection in Plain Dyed Fabric. *Sens. Mater.* **2021**, *33*, 4087–4103. [CrossRef]
20. Khodier, M.M.; Ahmed, S.M.; Sayed, M.S. Complex Pattern Jacquard Fabrics Defect Detection Using Convolutional Neural Networks and Multispectral Imaging. *IEEE Access* **2022**, *10*, 10653–10660. [CrossRef]

21. Siegmund, D.; Prajapati, A.; Kirchbuchner, F.; Kuijper, A. An Integrated Deep Neural Network for Defect Detection in Dynamic Textile Textures. In *Progress in Artificial Intelligence and Pattern Recognition, 6th International Workshop—IWAIPR 2018, Havana, Cuba, 24–26 September 2018*; Springer: Cham, Switzerland, 2018; pp. 77–84.
22. Yildiz, K.; Buldu, A.; Demetgul, M.; Yildiz, Z. A Novel Thermal-Based Fabric Defect Detection Technique. *J. Text. Inst.* **2015**, *106*, 275–283. [CrossRef]
23. Huang, T.S.; Schreiber, W.F.; Tretiak, O.J. Image Processing. *Proc. IEEE* **2005**, *59*, 1586–1609.
24. Kuruvilla, J.; Sukumaran, D.; Sankar, A.; Joy, S.P. A Review on Image Processing and Image Segmentation. In Proceedings of the 2016 International Conference on Data Mining and Advanced Computing (SAPIENCE), Ernakulam, India, 16–18 March 2016; pp. 198–203.
25. Di, L.; Long, H.; Liang, J. Fabric Defect Detection Based on Illumination Correction and Visual Salient Features. *Sensors* **2020**, *20*, 5147. [CrossRef]
26. Mewada, H.; Pires, I.M.; Engineer, P.; Patel, A.V. Fabric Surface Defect Classification and Systematic Analysis Using a Cuckoo Search Optimized Deep Residual Network. *Eng. Sci. Technol. Int. J.* **2024**, *53*, 101681. [CrossRef]
27. Hu, G.-H.; Wang, Q.-H.; Zhang, G.-H. Unsupervised Defect Detection in Textiles Based on Fourier Analysis and Wavelet Shrinkage. *Appl. Opt.* **2015**, *54*, 2963–2980. [CrossRef]
28. Kuo, C.F.J.; Wang, W.R.; Barman, J. Automated Optical Inspection for Defect Identification and Classification in Actual Woven Fabric Production Lines. *Sensors* **2022**, *22*, 7246. [CrossRef] [PubMed]
29. Jiang, J.; Jin, Z.; Wang, B.; Ma, L.; Cui, Y. A Sobel Operator Combined with Patch Statistics Algorithm for Fabric Defect Detection. *KSII Trans. Internet Inf. Syst. (TIIS)* **2020**, *14*, 687–701. [CrossRef]
30. Zhang, Y.F.; Bresee, R.R. Fabric Defect Detection and Classification Using Image Analysis. *Text. Res. J.* **1995**, *65*, 1–9. [CrossRef]
31. Jun, X.; Wang, J.; Zhou, J.; Meng, S.; Pan, R.; Gao, W. Fabric Defect Detection Based on a Deep Convolutional Neural Network Using a Two-Stage Strategy. *Text. Res. J.* **2021**, *91*, 130–142. [CrossRef]
32. Wang, P.H.; Lin, C.C. Data Augmentation Method For Fabric Defect Detection. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics—Taiwan, Taipei, Taiwan, 6–8 July 2022; pp. 255–256.
33. Muchlis, A.; Wibowo, E.P.; Irawan, R.; Afzeri. The Effect of Data Augmentation on Accuracy Values In Fabric Defect Detection. *Rev. Gestão-RGSA* **2025**, *19*, e011606. [CrossRef]
34. Lu, B.; Zhang, M.; Huang, B. Deep Adversarial Data Augmentation for Fabric Defect Classification With Scarce Defect Data. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 1–13. [CrossRef]
35. Guyon, I.; Elisseeff, A. An Introduction to Feature Extraction. In *Feature Extraction: Foundations and Applications*; Guyon, I., Nikravesh, M., Gunn, S., Zadeh, L.A., Eds.; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1–25, ISBN 978-3-540-35488-8.
36. Mutlag, W.K.; Ali, S.K.; Aydam, Z.M.; Taher, B.H. Feature Extraction Methods: A Review. *J. Phys. Conf. Ser.* **2020**, *1591*, 012028. [CrossRef]
37. Chen, M.; Yu, L.; Zhi, C.; Sun, R.; Zhu, S.; Gao, Z.; Ke, Z.; Zhu, M.; Zhang, Y. Improved Faster R-CNN for Fabric Defect Detection Based on Gabor Filter with Genetic Algorithm Optimization. *Comput. Ind.* **2022**, *134*, 103551. [CrossRef]
38. Rasamoelina, A.D.; Adjailia, F.; Sinčák, P. A Review of Activation Function for Artificial Neural Network. In Proceedings of the 2020 IEEE 18th World Symposium on Applied Machine Intelligence and Informatics (SAMI), Herlany, Slovakia, 23–25 January 2020; pp. 281–286.
39. Agarap, A.F. Deep Learning Using Rectified Linear Units (ReLU). *arXiv* **2019**, arXiv:1803.08375v2. [CrossRef]
40. Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 6–11 July 2015; Volume 37, pp. 448–456.
41. Gholamalinezhad, H.; Khosravi, H. Pooling Methods in Deep Neural Networks, a Review. *arXiv* **2020**, arXiv:2009.07485. [CrossRef]
42. Song, W.; Lang, J.; Zhang, D.; Zheng, M.; Li, X. Textile Defect Detection Algorithm Based on the Improved YOLOv8. *IEEE Access* **2025**, *13*, 11217–11231. [CrossRef]
43. Ozek, A.; Seckin, M.; Demircioglu, P.; Bogrekci, I. Artificial Intelligence Driving Innovation in Textile Defect Detection. *Textiles* **2025**, *5*, 12. [CrossRef]
44. Shumin, D.; Zhoufeng, L.; Chunlei, L. AdaBoost Learning for Fabric Defect Detection Based on HOG and SVM. In Proceedings of the 2011 International Conference on Multimedia Technology, Hangzhou, China, 26–28 July 2011; pp. 2903–2906.
45. Deotale, N.T.; Sarode, T.K. Fabric Defect Detection Adopting Combined GLCM, Gabor Wavelet Features and Random Decision Forest. *3D Res.* **2019**, *10*, 5. [CrossRef]
46. Dhivya, D.; Renuka Devi, M. Detection of Structural Defects in Fabric Parts Using a Novel Edge Detection Method. *Comput. J.* **2018**, *62*, 1036–1043. [CrossRef]
47. Yang, R.; Guo, N.; Tian, B.; Wang, J.; Liu, S.; Yu, M. Fabric Defect Detection via Saliency Model Based on Adjacent Context Coordination and Transformer. *J. Eng. Fibers Fabr.* **2024**, *19*, 15589250241258272. [CrossRef]

48. Luo, Q.; Fang, X.; Su, J.; Zhou, J.; Zhou, B.; Yang, C. Automated Visual Defect Classification for Flat Steel Surface: A Survey. *IEEE Trans. Instrum. Meas.* **2020**, *69*, 9329–9349. [CrossRef]
49. Voronin, V.; Sizyakin, R.; Zhdanova, M.; Semenishchev, E.; Bezuglov, D.; Zelemskii, A. Automated Visual Inspection of Fabric Image Using Deep Learning Approach for Defect Detection. In Proceedings of the Automated Visual Inspection and Machine Vision IV, Online, 21–26 June 2021; Volume 11787, p. 117870P.
50. Edozie, E.; Shuaibu, A.N.; John, U.K.; Sadiq, B.O. Comprehensive Review of Recent Developments in Visual Object Detection Based on Deep Learning. *Artif. Intell. Rev.* **2025**, *58*, 277. [CrossRef]
51. Dhamal, S.K.; Joshi, C.; Chakrabarti, P.; Chouhan, D.S. An Automated Framework for Fabric Defect Detection in Textile Inspection. *J. Inf. Syst. Eng. Manag.* **2025**, *10*, 353–367. [CrossRef]
52. Jia, Z.; Shi, Z.; Quan, Z.; Shunqi, M. Fabric Defect Detection Based on Transfer Learning and Improved Faster R-CNN. *J. Eng. Fibers Fabr.* **2022**, *17*, 15589250221086647. [CrossRef]
53. Liu, Z.; Liu, S.; Li, C.; Ding, S.; Dong, Y. Fabric Defects Detection Based on SSD. In Proceedings of the 2nd International Conference on Graphics and Signal Processing; Association for Computing Machinery, Sydney, NSW, Australia, 6–8 October 2018; pp. 74–78.
54. Liu, S.; Huang, L.; Zhao, Y.; Wu, X. Lightweight Single Shot Multi-Box Detector: A Fabric Defect Detection Algorithm Incorporating Parallel Dilated Convolution and Dual Channel Attention. *Text. Res. J.* **2024**, *94*, 209–224. [CrossRef]
55. Xie, H.; Zhang, Y.; Wu, Z. An Improved Fabric Defect Detection Method Based on SSD. *AATCC J. Res.* **2021**, *8*, 181–190. [CrossRef]
56. Hussain, M. YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection. *Machines* **2023**, *11*, 677. [CrossRef]
57. Mao, M.; Hong, M. YOLO Object Detection for Real-Time Fabric Defect Inspection in the Textile Industry: A Review of YOLOv1 to YOLOv11. *Sensors* **2025**, *25*, 2270. [CrossRef]
58. Khanam, R.; Hussain, M. What Is YOLOv5: A Deep Look into the Internal Features of the Popular Object Detector. *arXiv* **2024**, arXiv:2407.20892. [CrossRef]
59. Pereira, F.; Lopes, H.; Pinto, L.; Soares, F.; Vasconcelos, R.; Machado, J.; Carvalho, V. A Novel Deep Learning Approach for Yarn Hairiness Characterization Using an Improved YOLOv5 Algorithm. *Appl. Sci.* **2025**, *15*, 149. [CrossRef]
60. Lv, H.; Zhang, H.; Wang, M.; Xu, J.; Li, X.; Liu, C. Hyperspectral Imaging Based Nonwoven Fabric Defect Detection Method Using LL-YOLOv5. *IEEE Access* **2024**, *12*, 41988–41998. [CrossRef]
61. Terven, J.; Córdova-Esparza, D.M.; Romero-González, J.A. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1680–1716. [CrossRef]
62. Yaseen, M. What Is YOLOv8: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector. *arXiv* **2024**, arXiv:2409.07813.
63. Jin, Y.; Liu, X.; Nan, K.; Wang, S.; Wang, T.; Zhang, Z.; Zhang, X. A Real-Time Fabric Defect Detection Method Based on Improved YOLOv8. *Appl. Sci.* **2025**, *15*, 3228. [CrossRef]
64. YOLOv5 Benchmarks. Available online: https://github.com/ultralytics/yolov5#benchmarks] (accessed on 25 August 2025).
65. Pasam, S.K.; Bhosale, A.G. A Comprehensive Comparison Between YOLOv5 and YOLOv8 for Embedded Systems Applications. In Proceedings of the 2025 3rd International Conference on Smart Systems for Applications in Electrical Sciences (ICSSES), Tumakuru, India, 21–22 March 2025; pp. 1–6.
66. Khanam, R.; Hussain, M. Yolov11: An Overview of the Key Architectural Enhancements. *arXiv* **2024**, arXiv:2410.17725. [CrossRef]
67. Wu, T.; Xie, F. MSI-YOLO: An Efficient Model for Multiscale Fabric Defect Detection Based on Improved YOLOv11. *Text. Res. J.* **2025**, 00405175251329936. [CrossRef]
68. Chen, P.; Luo, Y.G.; Zhang, J.; Wang, B. PRD-YOLOv11: Efficient and Accurate Textile Tow Defect Detection via Progressive Representation Distillation. *Signal Image Video Process* **2025**, *19*, 671. [CrossRef]
69. Kang, X. Research on Fabric Defect Detection Method Based on Lightweight Network. *J. Eng. Fibers Fabr.* **2024**, *19*, 15589250241232153. [CrossRef]
70. Alqahtani, D.K.; Cheema, M.A.; Toosi, A.N. Benchmarking Deep Learning Models for Object Detection on Edge Computing Devices. In Proceedings of the International Conference on Service-Oriented Computing, Tunis, Tunisia, 3–6 December 2024.
71. Song, S.; Jing, J.; Huang, Y.; Shi, M. EfficientDet for Fabric Defect Detection Based on Edge Computing. *J. Eng. Fibers Fabr.* **2021**, *16*, 15589250211008346. [CrossRef]
72. Intel Realsense d435i Datasheet. Available online: https://www.intel.com/content/www/us/en/products/sku/190004/intel-realsense-depth-camera-d435i/specifications.html (accessed on 25 August 2025).
73. Jetson Orin Nano Developer Kit User Guide | NVIDIA Developer. Available online: https://developer.nvidia.com/embedded/learn/jetson-orin-nano-devkit-user-guide/index.html (accessed on 11 July 2025).

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.