Journal of Information Systems Engineering and Management
2025, 10(46s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

An Automated Framework for Fabric Defect Detection in Textile Inspection

Shital K Dhamal[1], Dr. Chandani Joshi[2], Prof. Prasun Chakrabarti[3], Lt. (Dr.) D.S. Chouhan[4]

[1] *Research Scholar, Faculty of Computing and Informatics, SPSU, Udaipur, India.*
[2] *Associate Professor, Faculty of Computing and Informatics, SPSU, Udaipur, India.*
[3] *Director, Directorate of Research & Publications, Dean International Affairs & Sr. Professor Faculty of Computing and Informatics, SPSU, Udaipur, India.*
[4] *Professor, Faculty of Data Science and Analytics, SPSU, Udaipur, India.*

ARTICLE INFO

Received: 26 Dec 2024

Revised: 14 Feb 2025

Accepted: 22 Feb 2025

ABSTRACT

Fabric quality is of paramount importance in the textile industry. As a result, defect detection in fabric is critical for quality assurance to rigorous standards. Traditional methods of fabric inspection can suffer from human error and be quite labor-intensive. Therefore, further automated systems are needed to support this process. This paper describes a novel deep learning, and machine vision framework using Detectron2, state-of-the-art computer vision package. The framework is based on the Mask R-CNN model for automatic fabric defect detection. The architecture uses a ResNeXt-101 backbone, Feature Pyramid Networks (FPN), and Region of Interest (RoI) Align to improve defect detection for multiple textures, and sizes of fabric. The model achieves excellent performance, with a mean average precision (mAP) between 93% and 97%, near 97% recall rate, and F1-scores of between 95%- 96%. This automated framework not only improves the fabric inspection process but also reduces human error. Therefore, reducing costs and providing product quality assurance. By incorporating deep learning into fabric defect detection, this research represents a significant advancement towards improving efficiencies, and quality control, across the textile industry.

Keywords: Fabric defect detection. Deep Learning, Mask R-CNN, Detectron2, Computer vision, Automated inspection, Textile quality control

I. INTRODUCTION

Fabric defect detection plays a crucial role in maintaining the quality of textile products, directly impacting production costs, customer satisfaction, and overall business efficiency. Traditionally, fabric inspection has been a manual process, carried out by skilled workers who inspect fabrics for defects such as cuts, holes, thread errors, and metal contamination. However, manual inspection is time-consuming, labor-intensive, and prone to errors due to human fatigue. As the textile industry embraces automation and smart manufacturing, there is a growing demand for systems that can accurately and efficiently detect fabric defects. Defect detection has been remarkably successful in recent years because of deep learning techniques, especially convolutional neural networks (CNNs). These methods have made it possible to develop systems that can recognize and categorize flaws on their own without requiring a lot of feature engineering. In applications like object recognition and segmentation, Mask R-CNN, a cutting-edge deep learning framework for instance segmentation, has demonstrated great promise. Its potential for detecting fabric defects has been shown in recent studies, providing advantages over manual inspection and simple image processing methods [1], [2], and [3].

The Detectron2 framework, a high-performance library created by Facebook AI Research based on Mask R-CNN, is the foundation for the fabric defect detection method we present in this study. Accurate fabric fault segmentation and classification are possible by utilizing Detectron2. In order to improve multi-scale feature identification and refinement, our system uses Region of Interest (RoI) pooling, Feature Pyramid Networks (FPN), and a ResNet50 backbone for robust feature extraction [4], [5], [6], [7]. Through experiments on both public and bespoke datasets, we assess the system's performance and demonstrate that the suggested method performs better than conventional approaches in terms of accuracy, resilience, and scalability.

Journal of Information Systems Engineering and Management
2025, 10(46s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/
Research Article

The paper is structured as follows. The paper is structured as follows. The related work in fabric flaw identification is reviewed in Section 2. The proposed methodology, data collection, labelling, preprocessing and augmentation is explained in Section 3. Model Architecture and training setup is given in Section 4. The experimental details and results are presented in Section 5. Section 6 includes performance comparison and wraps up the study and makes recommendations for future research directions.

II. RELATED WORK

In order to guarantee the consistency and quality of textiles during the manufacturing process, fabric defect detection is essential. The majority of traditional defect detection methods relied on labor-intensive, human error-prone manual inspection or simple image processing techniques. However, more automated and effective defect detection systems are now possible thanks to developments in machine learning, especially deep learning.

2.1 Traditional Methods and Early Machine Learning Approaches

Early methods for fabric defect detection were based on traditional image processing techniques. These methods employed edge detection algorithms like Sobel and Canny edge detectors, Gabor filtering, and texture analysis to detect fabric defects such as holes, stains, and surface irregularities. However, these techniques often struggled to detect subtle defects or handle the complex textures of fabrics [8][9].

Detecting defects in yarn-dyed fabrics can be quite a challenge because of their intricate layers and patterns [10]. One effective strategy involves blending texture analysis with pattern matching to spot issues like oil stains, holes, or misplaced threads, and this approach has shown impressive accuracy. Another GLCM-based technique looks at how pixel patterns shift across the fabric when viewed from different angles and window sizes [11]. While these methods can be successful, they do have their limitations—they can be sensitive to noise, demand a lot of computational power, and struggle with high-resolution images. Plus, they sometimes overlook details, especially when the fabric's texture is particularly complex or large-scale.

Enhancing methods for detecting fabric defects has been the focus of recent study. In order to extract characteristics from fabric and aid distinguish flaws and lower false alarms, a unique method combining morphological filters (MF) with a pretrained Gabor Wavelet Network (GWN) was presented [12]. 78 photos were used to evaluate this approach, and a low-cost inspection device was used for real-time analysis. To identify flaws in fine gray plain weave fabric, a hybrid method that combines morphological and correlational methodologies was developed, which successfully lowers false alarm rates [13]. Even though both approaches demonstrated increased detection accuracy, there are still issues to be resolved, like choosing the right defect model and controlling the selection of repeating elements. Although these developments show progress in detecting fabric defects, they also point to areas that still require improvement.

The proposed approach takes inspiration from biological vision and uses low-rank representation (LRR) to break down fabric images into two parts: clear background areas and noticeable defect regions [14]. It treats the normal fabric as a low-rank structure and the defects as sparse anomalies [15][16]. Essentially, the image is seen as a combination of these two parts. Instead of using the traditional SVD for dimensionality reduction, it applies eigenvalue decomposition on smaller blocks of the image, which makes it simpler to implement and effective when the image has good contrast. That said, there are a few drawbacks. The method doesn't perform well on low-contrast images. It also assumes that all defects are sparse and all backgrounds are low-rank, which might not work for every type of fabric. Breaking the image into blocks can introduce some artifacts, and the processing can still be a bit heavy for large images.

Machine learning techniques, such as Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), and Decision Trees, were introduced to improve the detection accuracy by learning from image features. While these methods showed improvement over traditional techniques, they were still dependent on manually extracted features (e.g., texture, color, shape), making them computationally expensive and requiring domain expertise [17][18]. Despite their effectiveness in some cases, these models were limited by their reliance on handcrafted features and struggled to scale efficiently for larger datasets or complex defect types.

2.2 Deep Learning Approaches to Fabric Defect Detection

Since the inception of Convolutional Neural Networks (CNNs), fabric defect detection, in fact, has undergone a paradigm shift. CNNs can automatically learn hierarchical features from raw image data while eliminating the need for manual feature extraction. This aspect has greatly enhanced and brought about extraordinary detection of different fabric defects such as cuts, holes, and thread errors [19][20]. Deep learning has really transformed the way we detect fabric defects, allowing for automated and precise inspections. Convolutional Neural Networks (CNNs) have taken over from traditional methods by learning patterns straight from raw images, which has significantly improved our ability to spot issues like holes and thread errors [21]. Deep learning, particularly with CNN-based models, has really revolutionized how we detect fabric defects in the textile industry, making the process much more automated and precise. Take models like YOLO-SCD, for instance; they cleverly combine attention mechanisms with depth-wise separable convolutions, which has significantly boosted both feature extraction and detection speed. In fact, YOLO-SCD has outshone YOLOv4 when it comes to accuracy and speed [22]. That said, there are still hurdles to overcome. For starters, finding large, labeled datasets can be quite a challenge and often comes with a hefty price tag. Plus, these models can struggle to maintain consistent performance across various fabric types or under changing conditions. The need for substantial computational power also complicates real-time applications, and the "black box" aspect of deep learning models can lead to concerns about trust and transparency. Sometimes, they might overlook subtle or overlapping defects, and to keep them effective, regular updates are necessary. Nevertheless, the potential of deep learning to transform fabric defect detection is truly exciting [23].

Recently, deep learning has really taken off as a go-to solution for detecting fabric defects, and the results have been quite impressive [23, 24]. It's playing a significant role in enhancing both product quality and production efficiency within the textile industry [25]. However, while deep learning shines in areas like image segmentation and classification, applying it on actual production lines still presents some challenges [26]. One major hurdle is that textile factories require algorithms that can operate in real time, which means they need to be incredibly fast and efficient. Additionally, gathering enough images of real defects is much trickier than collecting defect-free samples, making it difficult to train deep learning models effectively [27] [28].

Deep learning object detectors fall into two types: one-stage and two-stage models [29]. One-stage detectors are faster but tend to be less accurate, while two-stage detectors are more precise but slower. In fabric defect detection, the same trade-off applies. So, choosing the right model really comes down to finding the best balance between speed and accuracy for the task at hand. One-stage detection algorithms treat every location in an image as a potential object, which speeds up the process but can sometimes lead to missing finer defects. The SSD model, a CNN-based approach, works well for fabric defect detection [30], but its focus on speed means it may overlook subtle flaws. Ouyang et al. [31] introduced a CNN method for on-loom fabric inspection that addresses data imbalance with a dynamic activation layer, but it can still struggle with detecting more complex defects. DCNNs, like those from Liu et al. [32] and Zhou et al. [33], improve texture recognition and feature extraction, though they can be resource-intensive. Lastly, models like D4Net [34] and PRAN-Net [35] optimize feature extraction for large-pattern fabrics, but their complexity can make them less suitable for real-time applications.

As with other two-stage detection algorithms like Faster R-CNN, the region proposals are first generated and then passed to a deep CNN for further predictions. Some modified versions of Faster R-CNN have been implemented for fabric defect detection, integrating both local and global defect recognition [36]. For fabric defects, GANs are quite effective because they can create realistic defect samples and adjust to various fabric textures; however, they have difficulty with the detection of rare defects [37]. To address this, WGANs with transfer learning have been used, increasing their efficacy on unbalanced datasets [38]. A model with visual long-short-term memory (VLSTM) has also been proposed, performing well in public dataset experiments [39]. Attention mechanisms have been utilized for the detection of more complex or blurry defects, enabling focus on the most relevant features [40]. While effective, these techniques are computationally expensive, limiting real-time applications. Some other techniques combine deep learning with non-deep approaches, such as adding a CNN with handcrafted features aimed at removing noise, which may help but increases the complexity and computational strength needed [41].

A thorough review of prior research highlights those traditional techniques for fabric defect detection such as edge detection, texture analysis, and early machine learning models often fall short when applied to real-world textiles due to their sensitivity to noise, reliance on manual feature design, and difficulty scaling to high-resolution or complex images [8][9][10][11]. While hybrid methods that combine filtering and statistical approaches have shown some improvements in accuracy, they remain computationally demanding and often struggle with low-contrast or subtle defects [13]. More recently, modern approaches have shifted toward data-driven methods that learn patterns directly from images, offering higher accuracy and automation potential [17][18]. However, even these advanced techniques face practical challenges, such as the need for large, labeled datasets, high computational resources, and inconsistent performance across different fabric types or lighting conditions [27][28]. To overcome these limitations, this study employs the Detectron2 framework, incorporating a ResNet-50 backbone, Feature Pyramid Network (FPN), ROI, and Mask R-CNN [42] [43] [44] [45]. ResNet-50 enables effective pattern extraction from complex textile surfaces while maintaining computational efficiency [46]. FPN improves the system's ability to detect both small and large defects by combining image features at multiple scales [47]. ROI Mask R-CNN adds the capability for pixel-level segmentation, allowing for precise identification and outlining of defect regions [48]. Together, these components form a practical and scalable system, and with Detectron2's modular design and support for real-time processing, the framework is well-suited for industrial fabric inspection applications [49].

III. PROPOSED METHODOLOGY

3.1 Data Collection

Data extraction is a fundamental step in the process of fabric defect detection, as it involves collecting suitable and relevant image data for accurate analysis. For this study, all fabric images were sourced from the MVTech fabric defect dataset and through self collection [50][51]. This particular dataset was selected due to its wide coverage of fabric types, defect categories, and image qualities, making it well-suited for developing a reliable and generalizable detection system. Prior to analysis, the dataset was carefully prepared through standard pre-processing techniques, including image resizing, data augmentation, and normalization. It contains high-resolution images that display both defect-free and defective regions across various fabric textures.

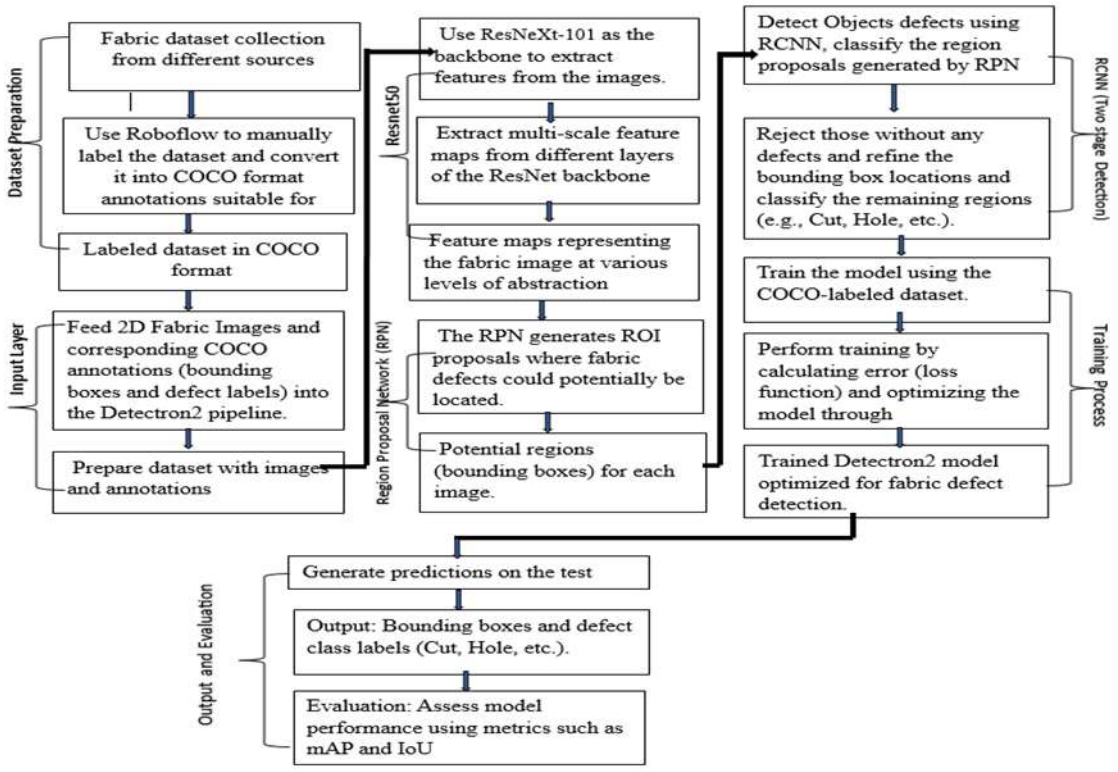
Dataset Preparation
Fabric dataset collection from different sources
Use Roboflow to manually label the dataset and convert it into COCO format annotations suitable for
Labeled dataset in COCO format
Input Layer
Feed 2D Fabric Images and corresponding COCO annotations (bounding boxes and defect labels) into the Detectron2 pipeline.
Prepare dataset with images and annotations
Resnet50
Use ResNeXt-101 as the backbone to extract features from the images.
Extract multi-scale feature maps from different layers of the ResNet backbone
Feature maps representing the fabric image at various levels of abstraction
Region Proposal Network (RPN)
The RPN generates ROI proposals where fabric defects could potentially be located.
Potential regions (bounding boxes) for each image.
RCNN (Two stage Detection)
Detect Objects defects using RCNN, classify the region proposals generated by RPN
Reject those without any defects and refine the bounding box locations and classify the remaining regions (e.g., Cut, Hole, etc.).
Training Process
Train the model using the COCO-labeled dataset.
Perform training by calculating error (loss function) and optimizing the model through
Trained Detectron2 model optimized for fabric defect detection.
Output and Evaluation
Generate predictions on the test
Output: Bounding boxes and defect class labels (Cut, Hole, etc.).
Evaluation: Assess model performance using metrics such as mAP and IoU

Figure 1: Proposed System for fabric defect detection using hybrid Detectron 2 architecture

To ensure the dataset aligned with the study's objectives, defect areas were manually annotated using pixel-level labeling tools. This manual process enabled precise identification of typical defects encountered in textile manufacturing, such as cuts, holes, thread inconsistencies, and contamination by foreign materials like metal. The detailed nature of these annotations significantly enhanced the accuracy and relevance of the dataset, making it suitable for both classification and segmentation tasks. As a result, the dataset served as a dependable benchmark for evaluating detection methods. This refined dataset ultimately contributed to the development of a more consistent and efficient approach to quality control in textile production.

3.2 Data Labelling

In the proposed methodology for fabric defect detection, Figure 1 presents the entire process flow, outlining all key steps from data collection to defect detection and classification.

Data labelling is one of the key steps in creating a machine learning-based pipeline where datasets with mindful annotations are used to teach models to learn and predict. This involves labelling each data point meaningfully so that the model has access to ground truth information from which patterns, relationships and decision boundaries can be found. High-quality labels are important, as they directly impact the model performance and reliability. Labelling assigns classes to data points (for example, using labels 'defective' and 'non-defective' for fabric defect detection) and may also include finer-grained attributes related to ground-truth defects (for example, types, locations, or severity levels of defects). Because of the significant role each label plays in model training, validation and testing, labelling is one of the most time-consuming and resource-intensive portions of ML model development as it often requires expertise, attention to detail, and domain knowledge. The dataset didn't have a COCO format annotation file suitable for Detecron2 architecture. Manual labelling is done to make images in Roboflow (e.g., JPG, PNG) compatible format, and an annotation file is created using Roboflow [50] [52] [53]. The dataset is organized into appropriate folders for each class (e.g., Cut, Hole, Thread Error, Metal Contamination) to keep the labelling process straightforward. Images are labelled manually by selecting the Labelling Interface within Roboflow. For each image, bounding boxes are drawn around the defect areas, and the appropriate class label (e.g., Cut, Hole, Thread Error, Metal Contamination) is assigned to the correct category. The resulting COCO JSON file is organized with the images, annotations, and categories sections following the correct structure.

3.3 Data Pre-processing and augmentation

Data pre-processing is the most essential stage in preparing the collected dataset for efficient fabric defect detection. It removes uncertainties such as noise, missing values, null values, and irrelevant data that impact the precision of classification. For the optimization of the dataset for training a model, we meticulously conduct the pre-processing phase. The image sizes in the dataset were resized to 640 x 640. Images are converted to grayscale by gray conversion, which simplifies processing and reduces computational complexity. Then resizing and rescaling techniques standardize the dimension and intensity values throughout the dataset. Gaussian filtering is used to remove unwanted noise that otherwise may hide defects. A Gaussian filter is a type of low-pass filter that helps reduce noise (high-frequency components) in an image, especially in areas that appear blurry. It smooths out edges and eliminates bright pixels. The filter operates by applying a symmetric kernel of odd size to each pixel within the Region of Interest (ROI), achieving the intended effect. The values in the kernel are determined using a 3×3 Gaussian 2D Filter with Standard Deviation = 1 [54] [55].

Finally, normalization techniques, such as min-max scaling or standardization, ensure uniform pixel values between 0 and 1 to ease the convergence of the model and eliminate feature dominance. These pre-processing techniques transform the raw dataset into a refined, consistent, and high-quality input for our deep learning model to increase the accuracy and dependability of defects in detection as shown in Figure 2

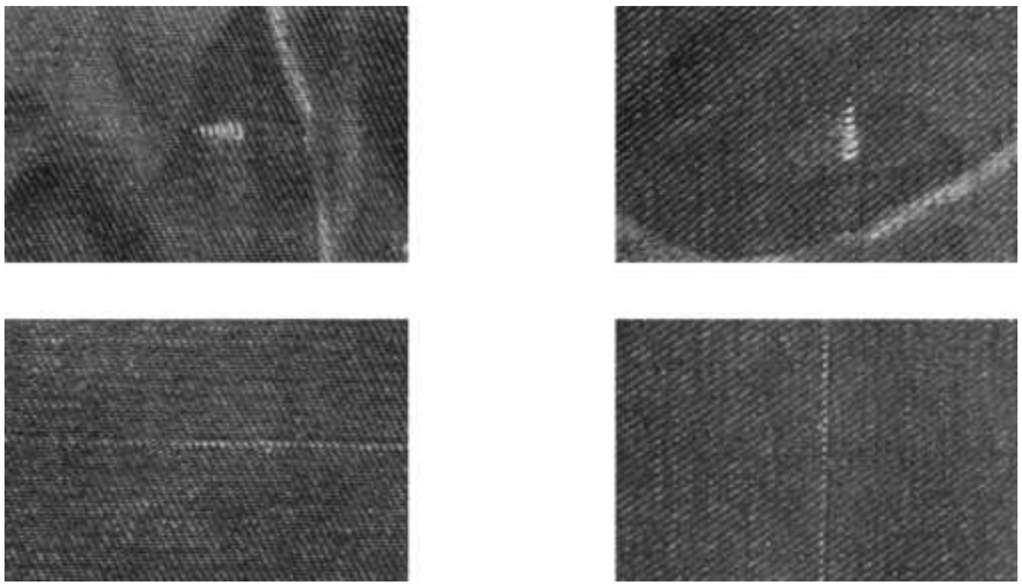

Figure 2: Visualization of preprocessed fabric images with annotations

As a means of improving the diversity and the number of images in the dataset, different techniques of data augmentation were used. These include augmentation in image flipping (horizontal flip, vertical flip, rotation ±15°, shear in horizontal and vertical direction angle of 0-20 degrees), hue shifts and brightness adjustment, and even image degradation (1-pixel blur). These augmentations introduced randomness in already existing images in a systematic way and thus led to an increase in the size of the datasets while increasing the generalizability of the model. The resulting dataset consists of 2749 images for training and 246 images for validation, thus it allows our model to better generalize out of sample unseen data and withstand variability of factors, as well as overall accuracy in fabric defect detection. Introducing augmented images helped mitigate the overfitting phenomenon and enhance the model's defect detection capability in several conditions, which improved the accuracy as well as the reliability of the defect detection system.

IV. MODEL ARCHITECTURE AND TRAINING SETUP

The fabric defect detection model is based on Detectron2, a state-of-the-art object detection and segmentation library developed by Facebook AI Research (FAIR). Detectron2 offers high performance, speed, and simplicity, making it ideal for fabric defect detection tasks, including instance segmentation and key point detection. The framework is built on the PyTorch platform and utilizes CUDA for fast computation, supporting large-scale computer vision applications. The schematic architecture of Detectron2 is shown in Figure 3 [7][42]. The model leverages a hybrid architecture that consists of multiple stages for fabric defect detection, as outlined below in Figure 3

ResNeXt-101 Backbone: The analysis of fabric images begins with the use of ResNeXt-101, a deep convolutional neural network that excels at extracting visual features across different levels of detail. It starts by identifying basic elements such as edges and textures, then gradually picks up more complex and abstract patterns. Unlike traditional architectures, ResNeXt-101 introduces a grouped convolution strategy, which allows it to process information through multiple parallel paths. This design boosts its capacity to recognize intricate features while maintaining computational efficiency. As a result, it’s particularly effective for tasks that require a nuanced understanding of visual content, such as identifying detailed characteristics in fabric textures [43] [46] [58].

Feature Pyramid Network (FPN): The features extracted by ResNet-50 are passed to a Feature Pyramid Network, which generates multi-resolution feature maps to detect fabric defects of varying sizes. The FPN defines pyramid levels as P2, P3, P4, P5, and P6, with each level focused on detecting defects of different sizes. P2 focuses on small defects (e.g., pinholes), P3 and P4 focus on medium-sized defects, and P5 and P6 focus on large defects (e.g., tears or stains) [44] [47] [58].

Region Proposal Network (RPN): The feature maps from the backbone network are analyzed by the RPN, which generates candidate bounding boxes or region proposals. These boxes represent regions likely containing fabric defects and are assigned objectness scores to estimate the probability of defect presence [56].

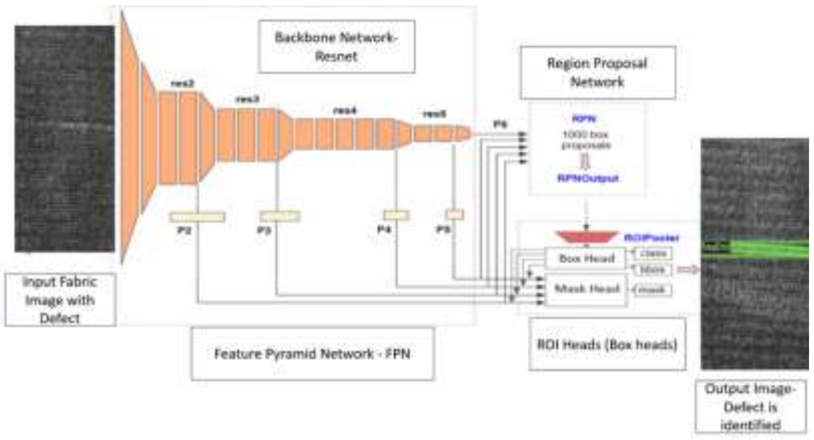

Figure 3: Schematic architecture of Detectron2 (redrawn figure based on https://github.com/ facebookresearch/detectron2

Mask RCNN and RoI Pooling: Mask R-CNN is an advanced method for detecting objects in images, such as fabric defects, by not only identifying them but also generating precise masks around them. When used for fabric defect detection, it can highlight issues like tears or holes. ROI pooling is crucial in this process, allowing the model to focus on various regions of the fabric and extract important features, regardless of size. This makes sure even minor defects are caught accurately. Together, Mask R-CNN and ROI pooling work seamlessly to identify fabric defects with great precision. The Region of Interest (RoI) Pooling step converts the proposed regions from the RPN into fixed-size feature maps, enabling uniform classification regardless of the regions' original sizes or aspect ratios [45]. Non-Maximum Suppression (NMS): NMS is applied to remove overlapping bounding boxes representing the same defect, retaining only the most confident predictions [57].

4.1 Training Setup

Experiment was implemented on Google Collab which uses TESLA T4 GPU (16GB VRAM). The model was fine-tuned using the pre-trained Faster R-CNN architecture, specifically the faster_rcnn_X_101_32x8d_FPN_3x variant that was originally trained on the COCO dataset. It uses a ResNet backbone with Bottleneck Blocks for better efficiency, and the Feature Pyramid Network helps capture features at different scales. To keep training stable, FrozenBatchNorm2d layers were used, and the Region Proposal Network relied on 3x3 convolutions to generate object proposals. The final layer was set up to detect five classes—four types of defects and one background class—with bounding box regression. Training ran for 1000 iterations with a batch size of 2 and a base learning rate of 0.00025. Learning rate decay was turned off, and the ROI batch size was set to 256. A custom Training Sampler and dataset serialization helped speed up data loading. When loading the pre-trained model, some typical warnings appeared due to differences between the COCO-trained weights and the custom dataset setup, especially in the ROI heads, which required some manual tweaking. The learning rate increased linearly from 5×10^{-6} to around 2.5×10^{-4}. The full training took about 19 minutes and used up to 3896 MB of GPU memory. Over time, the model improved steadily—total loss dropped from 1.88 to 0.22, classification loss came down significantly, and bounding box regression loss reduced gradually. By the end, RPN losses were almost gone, showing strong performance in proposing regions.

V. EXPERIMENTAL DETAILS AND RESULTS

5.1 Training and Validation Results

The custom dataset focused on four fabric defect classes: cut, thread error, metal contamination, and hole. Data was split into "train" and "valid" sets, and training outputs were saved in the "1000ep_train" directory. In total, there were 2,748 images used across training and validation, with annotations covering 618 holes, 724 metal contaminants, 697 thread errors, and 733 cuts. Images were formatted in COCO style and preprocessed using a ResizeShortestEdge method, which adjusted the shortest side of each image to 800 pixels, keeping the largest side under 1333 pixels. Random flipping and other augmentations were applied to help the model generalize better. Although the training logs showed clear improvements, no official metrics like mean Average Precision (mAP) were calculated due to the lack of an evaluator during inference. Still, mAP was the intended way to measure how well the model detected defects. Going forward, using tools like Detectron2's COCOEvaluator would help provide a more complete evaluation.

5.2 Performance Evaluation Metrics for Fabric Defect Detection

To assess the performance of the fabric defect detection techniques on the labeled dataset, several key metrics were used: accuracy, recall, precision, and mean Average Precision (mAP). These metrics are essential for understanding how well the framework detects fabric defects and how accurately it predicts defect locations in relation to the ground truth annotations.

Precision :- Precision tells us how often the defect predictions are correct. In simple terms, it measures the percentage of true fabric defects (correctly detected defects) among all the defects identified. It's calculated as:

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

Where, True Positives (TP) are the fabric defects (Hole, Cut, Thread Error, Metal Contamination) that were correctly identified and False Positives (FP) are instances where a defect was incorrectly identified. The higher the precision, the better the framework is at avoiding false alarms and correctly identifying Recall

Recall:- It measures how well the framework detects every relevant fabric defect in the dataset. It answers the question: "Out of all the actual defects, how many were successfully detected?" It's calculated as:

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

Where False Negatives (FN) are defects that were missed (i.e., defects present but not detected). A higher recall value means that the framework is better at identifying all fabric defects and minimizing Mean Average Precision (mAP).

Mean Average Precision (mAP):- It evaluates the overall performance of detecting fabric defects across all categories. It calculates the average precision (AP) for each defect type and then computes the mean of these values. The AP for each defect type is based on the area under the precision-recall curve, which shows how precision and recall trade off at different detection thresholds. A higher mAP indicates better performance, meaning the framework consistently detects defects across all categories.

$$mAP = \left(\frac{1}{N}\right) \times \sum_{i=1}^{N} AP_i \quad (3)$$

Where, N- total number of object class

Accuracy:- Lastly, accuracy provides an overall measure of the framework's performance by comparing the number of correctly identified fabric defects to the total number of instances. It's calculated as:

$$Accuracy = \frac{Correct\ Predictions(TP+TN)}{Total\ Instances} \quad (4)$$

Where: True Positives (TP) are the fabric defects (Hole, Cut, Thread Error, Metal Contamination) correctly detected, True Negatives (TN) are the non-defective fabrics correctly identified as such, False Positives (FP) are non-defective fabrics incorrectly labeled as defective, False Negatives (FN) are defective fabrics mistakenly classified as non-defective.This metric gives a broad view of the framework's ability to correctly identify both defective and non-defective fabrics.

VI. RESULTS AND ANALYSIS

6.1 Performance Metrics by Class

Table 1 highlights how well the proposed framework performs in detecting different types of fabric defects. The system shows strong accuracy and consistency across all defect categories, including Hole, Cut, Thread Error, and Metal Contamination. Its mean Average Precision (mAP@50) scores range from 93% to 97%, reflecting high reliability in identifying these issues. For instance, it achieves an impressive 97% mAP for Cut, followed by 96% for Thread Error and 95% for Metal Contamination. The Recall values also remain high—97% for Hole, Thread Error, and Metal Contamination, and 93% for Cut. The F1-scores, which indicate how well the system balances precision and recall, fall between 95% and 96%, reinforcing overall robustness. These results clearly show that the framework can accurately detect and classify different types of fabric defects, making it reliable for practical, real-world applications. Even though the Cut defect has a slightly lower recall, it still performs strongly overall, with high mAP and F1-score values that confirm its dependability.

Table 1. Detection performance across fabric defect types using key evaluation metrics.

Class	mAP%	Recall%	F1-score%
Hole	0.93	0.97	0.95
Cut	0.97	0.93	0.95
Thread Error	0.96	0.97	0.96
Metal Contamination	0.95	0.97	0.96

Figure 4 shows how accuracy improves during training. It tracks two key metrics *class_accuracy* (in blue) and *foreground_class_accuracy* (in orange) across 3000 training steps. Both metrics start off low but quickly improve and eventually settle close to 1.0, indicating that the system becomes highly accurate as training progresses. Class accuracy stabilizes first, while foreground class accuracy takes a bit more time, suggesting the system needed longer to consistently identify the more detailed features. Still, by the end of training, both metrics demonstrate strong, reliable performance.

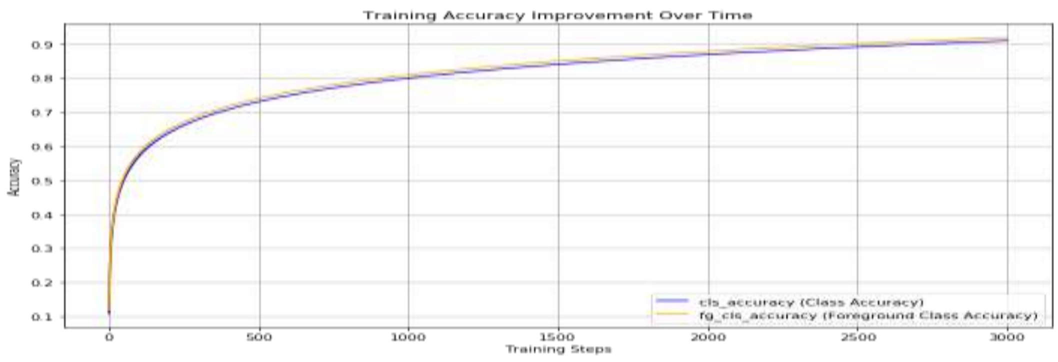

Figure 4: Training accuracy trends showing improvement in classification over time.

6.2 Training Dynamics and Loss Visualization

The system is trained to detect and classify common fabric defects, including Hole, Cut, Thread Error, and Metal Contamination. To achieve accurate detection and localization, a combined loss function is used, defined as [7]

$$L\_total = L\_rpn\text{-}cls + L\_rpn\text{-}box + L\_cls + L\_box \qquad (5)$$

In equation 5, each component of this loss contributes to a specific part of the detection process:

- L_rpn-cls: Classification loss from the region proposal stage, responsible for distinguishing between defect regions and non-defect background areas. Binary cross-entropy loss is applied here.
- L_rpn-box: Regression loss from the region proposal stage, used to fine-tune the bounding box coordinates around candidate defect areas. This term uses Smooth L1 loss.

- L_cls: Classification loss from the final prediction stage, ensuring that identified defects are correctly categorized into one of the defined classes (Hole, Cut, Thread Error, or Metal Contamination). Categorical cross-entropy loss is used.
- L_box: Final bounding box regression loss to enhance the precision of defect localization. This also uses the Smooth L1 loss function.

The Smooth L1 loss is calculated as:

$$\text{Smooth}_{L1}(x) = \begin{cases} 0.5 \cdot x^2, & \text{if } |x| < 1 \\ |x| - 0.5, & \text{otherwise} \end{cases} \tag{6}$$

This combined loss ensures accurate identification, classification, and localization of fabric defects, improving the reliability and efficiency of textile quality inspection systems.

The training process recorded the following metrics at regular intervals.

Table 2 - Training Loss Components and Learning Rate Across Iterations

Iteration	Total Loss	Loss cls	Loss class Reg	Loss RPN Cls	Loss RPN loc	Learning Rate
0	1.882	1.595	0.1542	0.1067	0.01013	5e-6
199	0.642	0.2834	0.3467	0.001569	0.005625	4.995e-5
499	0.5788	0.2137	0.3561	0.0004922	0.004804	0.00012488
799	0.372	0.1198	0.2271	1.194e-5	0.004155	0.0001998
999	0.2163	0.06058	0.1402	8.043e-7	0.004683	0.00024975

The Table 2 shows the evaluation results of the Faster R-CNN model on training data using the Detectron2 framework. The key performance metrics were tracked across 1000 iterations to monitor progress. Table 2 displays these results, showing how the model's total loss, classification loss, bounding box regression loss, and Region Proposal Network (RPN) losses evolved as training progressed. The learning rate started at 5e-6 and gradually increased to 0.00024975 by the end of the training, allowing the model to fine-tune its predictions and gradually improve. From the data in the table, we can see that the total loss dropped significantly from 1.882 at iteration 0 to 0.2163 at iteration 999, indicating steady learning and optimization. The classification loss saw the most notable improvement, reducing from 1.595 at the start to 0.06058 by the end, which highlights the growing ability to identify objects correctly. Additionally, the losses associated with bounding box regression and the RPN also decreased, with the RPN classification loss nearing zero—suggesting a strong and consistent convergence of the model. The entire training process was completed in under 19 minutes, which means the training was not only effective but also efficient, processing each iteration in just over 1 second on average. To further support accuracy, a resizing augmentation technique (ResizeShortestEdge) was applied, ensuring that the images remained consistent in scale without distortion, which improved the overall performance.

The Figure 5 illustrates the model's progress throughout the training process. There is a clear downward trend in the total loss, indicating consistent improvement. The most significant drop is seen in the classification loss, showing that the model became more accurate at recognizing objects. Losses related to object location and region detection also decreased steadily, reflecting a growing precision in identifying and localizing targets. Overall, the training followed a stable and efficient course, without any irregular spikes or setbacks.

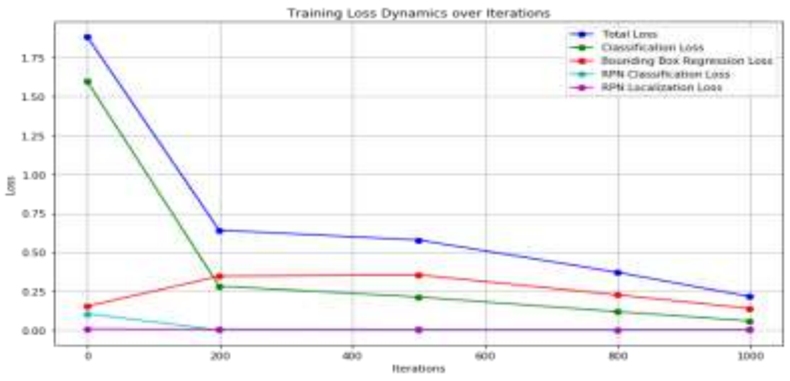

Figure 5: Training Loss Dynamics over Iterations

6.3 Fabric Defect Detection Outputs

The fabric images in Figure 6 - (a), (b), (c) and (d) shows different types of fabric defects detected and highlighted by the proposed model. The defect in each fabric image is highlighted by the surrounding box. The image (a) shows a thread error, which shows irregularity in the thread, caused by the slight misalignment in the weave. The image (b) shows a Hole error, which is caused due to weak yarns, friction, machine problem or during weaving or handling process. he model has identified cut or tear in the fabric caused by machine fault or accidental damage during weaving process as shown in Figure (c). It detects the presence of metal particles in the fabric image as shown in Figure (d) , such as broken needles, staples, or machine fragments that may have accidentally become embedded during the manufacturing process.

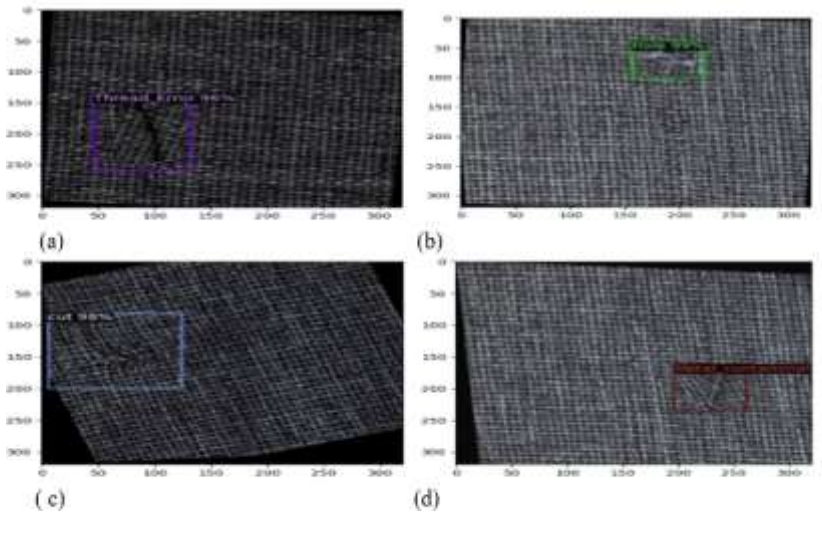

Figure 6: (a), (b), (c), (d) - Visualization of predicted defects

6.4 Effectiveness in Fabric Defect Detection

Table 3 Performance Comparison of Fabric Defect Detection Models

Model	Backbone	mAP(%)	Recall(%)
Proposed Model	ResNet-101	96.45	97.75
Improved Faster R-CNN with FPN, Soft-NMS, and ROI Align[58]	ResNet-101	82.6	85.0
DCFE-YOLO[58]	Dynamic Snake Convolution	89.4	83.7
Fab-ME[60]	YOLOv8s + C2F-VMamba + EMCA	59.4%	-
YOLOv8n[61]	YOLOv8n (nano)	84.8	83.9
YOLOv5n[61]	YOLOv5n (nano)	84.5	-
MobileNetSSD-FPNLite[61]	MobileNet + FPNLite	77.09%	-

The fabric defect detection results show a clear trend of improvement throughout the training process. Total loss and classification loss consistently decreased, indicating steady progress in learning. An initial increase in classification loss was observed, suggesting opportunities for refining the early stages of training. Loss values eventually stabilized, with total loss converging around 0.3, reflecting effective handling of both classification and localization tasks. Classification accuracy also improved over time. Both *cls_accuracy* and *fg_cls_accuracy* approached values near 1.0. The faster increase in *fg_cls_accuracy*, despite slight oscillations, along with the gradual improvement in *cls_accuracy*, points to progressive refinement in classification performance. Further support comes from additional evaluation metrics. Mean Average Precision (mAP) ranged from 93% to 97%, with the highest score observed for the "Cut" defect category. Recall reached up to 97%, indicating a low rate of missed defects, and F1-scores remained between 95% and 96%, showing a strong balance between precision and recall. As shown in Table 3, this approach achieved the highest overall performance, with an mAP of 96.45% and Recall of 97.75%. In comparison, other techniques using the same backbone—such as Faster R-CNN with FPN, Soft-NMS, and ROI Align—reached only 82.6% mAP and 85.0% Recall. Additional methods, including DCFE-YOLO, YOLOv8n, YOLOv5n, Fab-ME, and MobileNetSSD-FPNLite, displayed lower results in mAP, Recall, or both. These outcomes demonstrate a clear advantage in accuracy and consistency across various defect types as shown in Table 3 [58] [59] [60] [61].

VII. CONCLUSION AND FUTURE SCOPE

This study presents a practical and reliable framework for automated fabric defect detection using a high-performance image analysis system built on the Detectron2 platform. The approach integrates a ResNet-101 backbone with a Region Proposal Network (RPN) and Mask R-CNN to accurately identify, classify, and localize various types of defects found in textile materials. The training process demonstrated consistent progress, with the total loss decreasing from 1.882 at the beginning to 0.2163 at the final iteration. Classification loss saw the most substantial improvement, reflecting enhanced recognition capabilities over time. The full training run was completed in under 19 minutes, highlighting the system's overall efficiency and practicality for industrial use.

Evaluation results across multiple defect categories—including Hole, Cut, Thread Error, and Metal Contamination—showed high levels of accuracy and consistency. Mean Average Precision (mAP@50) scores ranged from 93% to 97%, recall values reached up to 97%, and F1-scores remained between 95% and 96%. Even for the Cut category, which showed slightly lower recall, the system still maintained strong overall performance.

Accuracy trends observed during the training process confirmed the framework's ability to consistently improve over time. Both class-level and foreground-level accuracy steadily increased, ultimately stabilizing near maximum values, indicating precise and reliable performance in identifying detailed features and defect boundaries.

Overall, the results confirm the system's suitability for real-world fabric inspection tasks, where both speed and accuracy are critical. Future efforts will focus on improving early-stage training behavior, extending support to a wider variety of fabric types and defect categories, and optimizing the solution for real-time operation within production environments.

REFERENCES

[1] Revathy, G., Kalaivani, R. Fabric defect detection and classification via deep learning-based improved Mask RCNN. SIViP 18, 2183–2193 (2024). https://doi.org/10.1007/s11760-023-02884-6

[2] X. Li, W. Cui, F. Jin and Q. Yu, "Fabric Linear Defect Detection Based on Mask RCNN," 2022 IEEE 6th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC), Beijing, China, 2022, pp. 1073-1079, doi: 10.1109/IAEAC54830.2022.9929606.

[3] He D, Wen J, Lai Z. Textile Fabric Defect Detection Based on Improved Faster R-CNN. AATCC Journal of Research. 2022;8(1_suppl):82-90. doi:10.14504/ajr.8.S1.11

[4] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. https://doi.org/10.1109/CVPR.2016.90

[5] K. He, G. Gkioxari, P. Dollár and R. Girshick, "Mask R-CNN," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 2980-2988, doi: 10.1109/ICCV.2017.322.

[6] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B., & Belongie, S. (2017). Feature pyramid networks for object detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2117–2125. https://doi.org/10.1109/CVPR.2017.106

[7] Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). *Detectron2*. https://github.com/facebookresearch/detectron2

[8] Divya, I. (2017, January). Detect the fabric defect area using edge detection techniques. International Journal of Computational Research and Development, Special Issue, 51–54

[9] Ding, G. (2020). Automatic detection of cloth defects based on Gabor filtering. In C. Huang, Y. W. Chan, & N. Yen (Eds.), Data processing techniques and applications for cyber-physical systems (DPTA 2019) (Vol. 1088, pp. [insert page numbers if known]). Springer. https://doi.org/10.1007/978-981-15-1468-5_69

[10] Zhu, D., Pan, R., Gao, W., & Zhang, J. (2015). Yarn-dyed fabric defect detection based on autocorrelation function and GLCM. Autex research journal, 15(3), 226-232.

[11] Zhang, X., & Fan, X. (2016, November). Fabric defect detection based on GLCM approach. In 6th International Conference on Information Engineering for Mechanics and Materials (pp. 673-677). Atlantis Press.

[12] Mak, K. L., Peng, P., & Yiu, K. F. C. (2009). Fabric defect detection using morphological filters. Image and Vision Computing, 27(10), 1585-1592.

[13] Jayashree, V., & Subbaramn, S. (2012, July). Hybrid Approach using correlation and morphological approaches for GFDD of plain weave fabric. In 2012 IEEE Control and System Graduate Research Colloquium (pp. 197-202). IEEE

[14] Li, C., Gao, G., Liu, Z., Yu, M., & Huang, D. (2018). Fabric defect detection based on biological vision modeling. *IEEE Access, 6*, 27659–27670.

[15] Li, P., Liang, J., Shen, X., Zhao, M., & Sui, L. (2019). Textile fabric defect detection based on low-rank representation. *Multimedia Tools and Applications, 78*(1), 99–124.

[16] Li, C., Li, J., Li, Y., He, L., Fu, X., & Chen, J. (2021). Fabric defect detection in textile manufacturing: a survey of the state of the art. Security and Communication Networks, 2021(1), 9948808.

[17] Suphongwibunphan R., Sangsuwan, T., Hansot, J., Wongsaroj, W., & Thong-un, N. (2024). A Novel Method for Knitted Fabric Defect Classification Using Image Processing and Weighted Voting classifiers. Journal Européen des Systèmes Automatisés, 57(4), 1047.

[18] Andre, A. B., Beltrame, E., & Wainer, J. (2013). A combination of support vector machine and k-nearest neighbors for machine fault detection. Applied Artificial Intelligence, 27(1), 36-49.

[19] Xiang, J., Zhang, N., Pan, R., & Gao, W. (2019). Fabric image retrieval system using hierarchical search based on deep convolutional neural network. Ieee Access, 7, 35405-35417.

[20] Ngan, H. Y., Pang, G. K., & Yung, N. H. (2011). Automated fabric defect detection—A review. *Image and vision computing*, *29*(7), 442-458.

[21] Liu, Q., Wang, C., Li, Y., Gao, M., & Li, J. (2022). A fabric defect detection method based on deep learning. IEEE Access, 10, 4284-4296. https://doi.org/10.1109/ACCESS.2021.3140118

[22] Luo, X., Ni, Q., Tao, R., & Shi, Y. (2023). A lightweight detector based on attention mechanism for fabric defect detection. *IEEE Access, 11*, 33554-33569. https://doi.org/10.1109/ACCESS.2023.3264262

[23] Rasheed, A., Zafar, B., Rasheed, A., Ali, N., Sajid, M., Dar, S. H., & Mahmood, M. T. (2020). Fabric defect detection using computer vision techniques: a comprehensive review. Mathematical Problems in Engineering, 2020(1), 8189403

[24] Siegmund, D., Fu, B., Jose-Garcia, A., et al. (2020). Study and research on detection of fiber defects using keypoints and deep learning. International Journal of Pattern Recognition & Artificial Intelligence.

[25] Das, S., Wahi, A., Keerthika, S., & Nulasiram, N. (2020). Defect analysis of textiles using artificial neural network. Current Trends in Fashion Technology & Textile Engineering, 6(1)

[26] Barua, S., Patil, H., Desai, P. D., et al. (2020). Deep learning-based smart colored fabric defect detection system. In Applied Computer Vision and Image Processing (pp. 212–219).

[27] Zhou, X., Li, Y., & Liang, W. (2020). CNN-RNN based intelligent recommendation for online medical pre-diagnosis support. IEEE/ACM Transactions on Computational Biology and Bioinformatics

[28] Li, Y., Zhao, W., & Pan, J. (2017). Deformable patterned fabric defect detection with Fisher criterion-based deep learning. IEEE Transactions on Automation Science and Engineering, 14(2), 1256–1264.

[29] Wu, X., Sahoo, D., & Hoi, S. C. H. (2020). Recent advances in deep learning for object detection. *Neurocomputing, 396*, 39–64. https://doi.org/10.1016/j.neucom.2019.11.031

[30] Liu, Z., Liu, S., & Li, C. (2018). Fabric defects detection based on SSD. In *Proceedings of the 2nd International Conference on Graphics and Signal Processing: ICGSP'18*, Sydney, Australia, October 2018.

[31] Ouyang, W., Xu, B., Hou, J., & Yuan, X. (2019). Fabric defect detection using activation layer embedded convolutional neural network. *IEEE Access, 7*, 70130–70140. https://doi.org/10.1109/ACCESS.2019.2918012

[32] Liu, Z., Zhang, C., Li, C., et al. (2019). Fabric defect recognition using optimized neural networks. *Journal of Engineered Fibers and Fabrics, 14*, 1-9. https://doi.org/10.1177/1558925019014001O1

[33] Zhou, T., Zhang, J., Su, H., Zou, W., & Zhang, B. (2021). EDDs: A series of Efficient Defect Detectors for fabric quality inspection. *Measurement, 172*, 108885. https://doi.org/10.1016/j.measurement.2020.108885

[34] Xu, X., Chen, J., Zhang, H., & Ng, W. W. Y. (2021). D4Net: De-deformation defect detection network for non-rigid products with large patterns. *Information Sciences, 547*, 763–776. https://doi.org/10.1016/j.ins.2020.09.022

[35] Peng, P., Wang, Y., Hao, C., Zhu, Z., Liu, T., & Zhou, W. (2020). Automatic fabric defect detection method using PRAN-net. *Applied Sciences, 10*(23), 8434. https://doi.org/10.3390/app10238434

[36] Jun, X., Wang, J., Zhou, J., & et al. (2020). Fabric defect detection based on a deep convolutional neural network using a two-stage strategy. Textile Research Journal, 91(1-2), 130–142.

[37] Liu, J., Wang, C., Su, H., & et al. (2020). Multistage GAN for fabric defect detection. IEEE Transactions on Image Processing, 29, 3388–3400.

[38] Le, X., Mei, J., Zhang, H., & et al. (2020). A learning-based approach for surface defect detection using small image datasets. Neurocomputing, 408, 112–120.

[39] Zhao, Y., Hao, K., He, H., & et al. (2020). A visual long-short-term memory based integrated CNN model for fabric defect image classification. Neurocomputing, 380, 259–270.

[40] Wang, J., Liu, Z., Li, C., & et al. (2020). Self-attention deep saliency network for fabric defect detection. communications in Computer and Information Science, 1160, Singapore.

[41] Wang, J., Li, C., Liu, Z., & et al. (2019). Combining deep and handcrafted features for NTV-NRPCA based fabric defect detection. In Proceedings of the 2019 Springer Cham.

[42] Wang, P., Zhang, L., & Zhang, X. (2021). Fabric defect detection using Detectron2 framework: An experimental study. arXiv preprint arXiv:2102.00414. https://arxiv.org/abs/2102.00414

[43] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770-778. https://doi.org/10.1109/CVPR.2016.90

[44] Lin, T. Y., Dollár, P., Girshick, R., He, K., & Hariharan, B. (2017). Feature pyramid networks for object detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2117-2125. https://doi.org/10.1109/CVPR.2017.106

[45] He, K., Gkioxari, G., Dollár, P., & Girshick, R. (2017). Mask R-CNN. Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2961-2969. https://doi.org/10.1109/ICCV.2017.324

[46] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated residual transformations for deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1492-1500).

[47] Xie, H., Zhang, Y., & Wu, Z. (2019). Fabric defect detection method combining image pyramid and direction template. IEEE Access, 7, 182320-182334. https://doi.org/10.1109/ACCESS.2019.2959880

[48] He, D., Wen, J., & Lai, Z. (2021). Textile fabric defect detection based on improved faster R-CNN. AATCC Journal of Research, 8(1_suppl), 82-90.

[49] Rani, A., Ortiz-Arroyo, D., & Durdevic, P. (2024). Defect detection in synthetic fibre ropes using detectron2 framework. Applied Ocean Research, 150, 104109

[50] Projects. (2024, March). mvtec-fabric-defect Dataset. Roboflow Universe. Roboflow. https://universe.roboflow.com/projects-nlys5/mvtec-fabric-defect

[51] Bergmann, P., Batzner, K., Fauser, M., & Steger, C. (2019). MVTec AD – A comprehensive real-world dataset for unsupervised anomaly detection. MVTec Software GmbH. https://www.mvtec.com/company/research/datasets/mvtec-ad

[52] Roboflow. (**n.d.**). *Roboflow: Annotate, preprocess, and manage computer vision datasets*. https://roboflow.com/

[53] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft COCO: Common objects in context. In D. Fleet, T. Pajdla, B. Schiele, & T. Tuytelaars (Eds.), European Conference on Computer Vision (ECCV 2014), Lecture Notes in Computer Science (Vol. 8693, pp. 740–755). ringer. https://doi.org/10.1007/978-3-319-10602-1_48

[54] OpenCV. (n.d.). cv2.GaussianBlur in OpenCV. Retrieved from https://docs.opencv.org/master/d4/d13/tutorial_py_filtering.html

[55] Gonzalez, R. C., & Woods, R. E. (2008). Digital Image Processing (3rd ed.). Prentice-Hall.

[56] Li, W., Zhang, Z., Wang, M., & Chen, H. (2023). Fabric defect detection algorithm based on image saliency region and similarity location. Electronics, 12(6), 1392.

[57] Liu, Q., Wang, C., Li, Y., Gao, M., & Li, J. (2022). A fabric defect detection method based on deep learning. IEEE Access, 10, 4284-4296. https://doi.org/10.1109/ACCESS.2021.3140118

[58] He D, Wen J, Lai Z. Textile Fabric Defect Detection Based on Improved Faster R-CNN. AATCC Journal of Research. 2022;8(1_suppl):82-90. doi:10.14504/ajr.8.S1.11

[59] Zhou L, Ma B, Dong Y, Yin Z, Lu F (2025) DCFE-YOLO: A novel fabric defect detection method. PLoS ONE 20(1): e0314525. https://doi.org/10.1371/journal.pone.0314525

[60] Wang, S., Kong, H., Li, B., & Zheng, F. (2024). Fab-ME: A Vision State-Space and Attention-Enhanced Framework for Fabric Defect Detection. arXiv preprint arXiv:2412.03200.

[61] Nasim, M., Mumtaz, R., Ahmad, M., & Ali, A. (2024). Fabric Defect Detection in Real World Manufacturing Using Deep Learning. Information, 15(8), 476.