IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Fashion Material Identification And Its Damage Detection Using Deep Learning.

Ramdas Bagawde[1], Rahul Gawade[2], Vrushali More[3], Sakshi Pawale[4], Vaishnavi Jadhav[5]

[1]Assi. Prof. Department of Comp. Engg., Govt College of Engg & Research Avasari Khurd, Pune, Maharashtra, India.

[2,3,4,5]UG Student, Department of Comp. Engg., Govt College of Engg & Research Avasari Khurd, Pune, Maharashtra, India.

Abstract: The fashion industry experiences problems with identification of fabrics and defects, leading to higher amounts of waste and inefficiency in quality control. This research explores an AI-driven approach, using Region-Based Convolutional Neural Networks (R-CNN) in fashion material classification and You Only Look Once (YOLO) for real-time damage detection. Combining deep learning with image processing will ensure high accuracy, automation, and scalability of the system during textile inspection. It improves quality control, reduces wastes, and encourages sustainable production. Results from experiments will show that it enhances both textile classification and defects detection, which makes the proposed solution an essential for the textile industry.

Keywords: Fabric Defect Detection, R-CNN, YOLO, Deep Learning, Image Processing

I. INTRODUCTION

The fashion industry faces significant challenges in fabric quality control, leading to waste generation due to defects, inaccurate material identification, and inefficient inspection processes. Manual inspections are time-consuming, error-prone, and lack scalability. To address these issues, this research introduces a deep learning-based solution that integrates image processing, machine learning, and automation for precise material identification and defect detection in textiles.

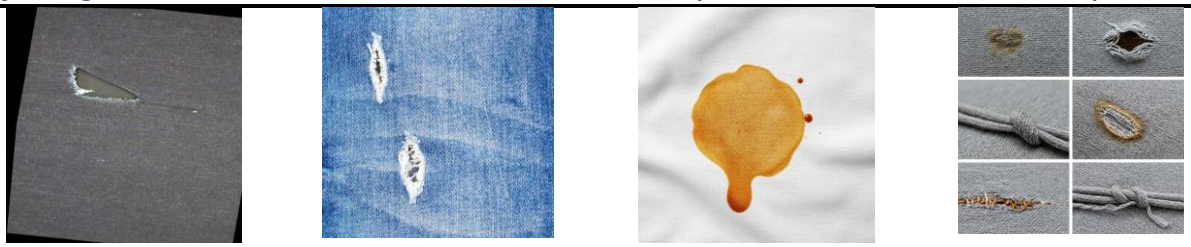
This study employs two state-of-the-art models: Region-Based Convolutional Neural Networks (R-CNN) for fabric classification and You Only Look Once (YOLO) for real-time damage detection. R-CNN effectively classifies textile materials, leveraging deep feature extraction to distinguish between various fabric types. YOLO enables rapid defect localization, identifying issues such as stains, holes, and weaving irregularities with high efficiency.

A. Data Collection:

Images of fabrics with various materials and defects are gathered from pre-collected datasets and real-time camera capture, including video when required. High-resolution cameras and OpenCV are used for image and video acquisition.

a) Dataset Overview:

We have taken the dataset from kaggle platform. This fabric defect database consists of 3 classes of defective images namely horizontal, stain, thread and holes along with 3 mask images for each defective image sample. Images have a size of 640x360. The dataset consists of images of different fabric materials with various defects like horizontal, vertical, holes, and line defects [1]. The dataset is split into 80% for training and 20% for testing to ensure optimal model learning and evaluation. These images are utilized for training the deep learning models to classify the material precisely and detect the defects. Some sample defect types are depicted below.



a) Hole b) Horizontal c) Stain d) Thread

Figure I Fabric Defects Images

B. Data Pre-processing:

Image quality is enhanced to improve model training. Techniques used include image resizing and normalization (OpenCV, NumPy), noise reduction (Gaussian Blur, Median Filtering), contrast enhancement (Histogram Equalization - CLAHE), and data augmentation (rotation, flipping, brightness adjustment using TensorFlow/Keras).

a) Data Cleaning:

In this process, no explicit method is used for handling missing data since images inherently do not contain missing values like tabular datasets. To manage noisy data, we utilize OpenCV's cv2.resize() function to standardize image dimensions, ensuring consistency across the dataset. Additionally, we configure a confidence threshold of 0.1 in TensorFlow's InferenceConfiguration, which helps filter out low-confidence detections, thereby improving the accuracy of fabric defect identification [2] [3].

b) Data Transformation:

Image resizing is performed using OpenCV's cv2.resize((800, 600)), ensuring that all images maintain a uniform size suitable for deep learning model training. To improve object localization, bounding box normalization is implemented using PIL (Pillow) with draw.rectangle(...), where bounding box coordinates are recalculated relative to object dimensions for better visualization. Furthermore, the deep learning model is defined using TensorFlow/Keras, where a structured input representation is created using layers such as tf.keras.layers.Dense(64, activation='relu', input_shape=(784,)), enabling efficient feature extraction and classification [4].

c) Data Reduction:

To reduce computational complexity while preserving key features, images are downscaled using OpenCV's cv2.resize((800, 600)). Inferencing is also improved by setting the confidence threshold (confidence_threshold=0.1) in TensorFlow so only high confidence predictions are included in the classification of fabric and defect detection. This helps streamline processing while maintaining model accuracy and efficiency [3] [2].

C. Feature Extraction:

Feature extraction is a crucial step in deep learning-based fabric defect detection, where meaningful patterns are identified from images to enhance model performance. Unlike traditional machine learning, which requires manual feature selection, deep learning models automatically extract relevant features from images using convolutional neural networks (CNNs). In this project, feature extraction is performed through edge detection and deep feature mapping techniques to improve material classification and defect detection.

a) *Edge Detection:*

To highlight fabric patterns and defect boundaries, we employ Canny Edge Detection, implemented using OpenCV's cv2.Canny() function. This technique enhances defect visibility by detecting high-gradient regions, allowing the model to distinguish between fabric textures and anomalies like stains, tears, and holes [5] [6].

b) *Deep Feature Extraction:*

High-level feature representation is achieved using convolutional feature maps generated by CNNs. The model automatically extracts hierarchical features such as texture, shape, and defect patterns. We use pre-trained deep learning architectures in TensorFlow and PyTorch, where convolutional layers apply filters to learn edge patterns, color distributions, and structural differences in fabrics [7].

c) *Feature Normalization:*

To ensure consistency in extracted features, normalization is performed using Batch Normalization (tf.keras.layers.BatchNormalization()). This stabilizes training by reducing internal covariate shifts, ensuring uniform feature distribution across different fabric samples [8].

D. Model Training:

Model training is a crucial phase where deep learning models learn to classify different fabric materials and detect defects in textile images. In this project, training is performed using Faster R-CNN for fabric classification and YOLO for real-time defect detection. These models are trained on a labeled dataset containing images of different fabric types and defects, ensuring effective learning and generalization.

a) Dataset Splitting:

The Fabric Defect Dataset from Kaggle is split into 80% for training and 20% for testing, ensuring a balanced distribution for model learning [9]. This allows the model to learn from a diverse set of images while retaining an unseen portion for evaluation.

b) Optimization Techniques:

The training process is optimized using the Adam optimizer, which dynamically adjusts learning rates for faster convergence. Additionally, learning rate scheduling is applied to prevent overfitting and enhance model stability during training [10].

c) Loss Functions:

The cross-entropy loss function is implemented for fabric classification, and ensure correct classification of the different types of material. For defect detection, Smooth L1 loss is applied in Faster R-CNN to improve bounding box localization accuracy, while YOLO's loss function combines classification, localization, and confidence loss to enhance real-time defect detection [11].

d) Training Configuration:

The models are implemented using TensorFlow and PyTorch, utilizing Batch Normalization (tf.keras.layers.BatchNormalization()) to improve training stability and ensure consistent feature scaling. Image augmentation techniques, such as flipping, rotation, and brightness adjustments, are applied to improve the model's robustness [11].

e) Evaluation Metrics:

Model performance is assessed using key evaluation metrics, including accuracy, precision, recall, F1-score, and mean Average Precision (mAP). These metrics provide a detailed analysis of the model's effectiveness in both fabric classification and defect detection [9].

f) Hardware Utilization:

To accelerate computation and handle large-scale image processing, training is conducted on high-performance GPUs. This significantly reduces model training time and enhances the efficiency of real-time defect detection [10].

E. Model Testing & Deployment:

Model testing and deployment ensure the trained model performs effectively in real-world textile quality control.

a) *Model Testing:*

The model is evaluated on the 20% test dataset using accuracy, precision, recall, F1-score, and mean Average Precision (mAP) [9]. Cross-validation and confusion matrices help analyze misclassifications, ensuring robustness.

b) *Inference Performance Evaluation:*

The model's speed and efficiency are tested using Frames Per Second (FPS) and computational cost analysis. YOLO enables real-time defect detection, while Faster R-CNN ensures precise classification [10].

c) *Deployment Pipeline:*

The trained model is deployed via Flask (API development) and TensorFlow Serving for real-time inference. A React JS frontend enables user interaction for textile inspection.

d) *Real-Time Data Processing:*

The system processes live images and videos using OpenCV, allowing real-time fabric classification and defect detection with bounding boxes [11].

e) Scalability & Optimization:

Docker containerization enables easy deployment across environments, while GPU acceleration and TensorRT optimization enhance inference efficiency.

II. RELATED WORK

This section provides a review of available literature in fabric material classification and defect detection with deep learning. The papers are compared based on title, authors, algorithm/strategy employed, merits, and drawbacks.

Table 1 Comparing Previous Related Work

Paper Title	Authors	Algorithm/Strategy Used	Advantages	Disadvantages
Fabric Defect Detection Based on Cascade Faster R-CNN [12]	Zhiyong Zhao, Kang Gui, Peimao Wang (2020)	Cascade Faster R-CNN	Improved detection accuracy for various defect types	Increased computational complexity
Fabric Defect Detection Using the Improved YOLOv3 Model [13]	Dong Zhuo, Min Zheng, Huanhuan Zhang, Yong Li	Enhanced YOLOv3 with dimension clustering and K-means algorithm	Effective real-time defect detection; reduced error rates	May struggle with detecting very small defects

	ang (2018)			
Fabric Defect Detection and Classification via Deep Learning-Based Improved Mask R-CNN [14]	G. Revathy, R. Kalaivani (2023)	Improved Mask R-CNN	High precision in defect detection and classification	Requires substantial computational resources
A Fabric Defect Detection Method Based on Deep Learning [15]	Qiang Liu, Chuan Wang, Yusheng Li, Mingwan	YOLOv4 with Spatial Pyramid Pooling (SPP) and SoftPool	Improved detection accuracy, better feature extraction	Requires high computational power for training

	g Gao, Jingao Li (2020)			
Fabric Defect Detection in Real World Manufacturing Using Deep Learning [9]	Mariam Nasim, Rafia Mumtaz, Muneer Ahmad and Arshad Ali (2024)	Deep Learning model trained on real-world fabric datasets	High accuracy, relevant for real-world textile industry	Computational cost is a concern for large-scale deployment
Efficient Fabric Classification and Object	Makara Mao, Ahyoung Le	YOLOv10 for fabric classification and defect detection	Faster processing, enhanced acc	Requires specialized annotated dataset

Detection Using YOLOv10 [16]	e, Min Hong (2024)		ura cy for tex tile def ect ide nti fic ati on	s for opti mal per for ma nce

Research Gap and Contribution

Most of the currently available studies have either material classification or defect detection as their centre of focus but with limitations to real-time and accuracy. The current study merges R-CNN for material class and YOLO for defects, offering a real-time automatic, efficient, and scalable technique for textile quality inspection.

IV. R-CNN

Object detection is object identification and classification in an image. Among the deep learning techniques, region with convolutional neural networks (R-CNN) combines rectangular region proposals with the features of a convolutional neural network. R-CNN is a two-stage system for detection. The first detects a subset of regions in an image which may or may not contain an object. The second classifies the object in each region [17].

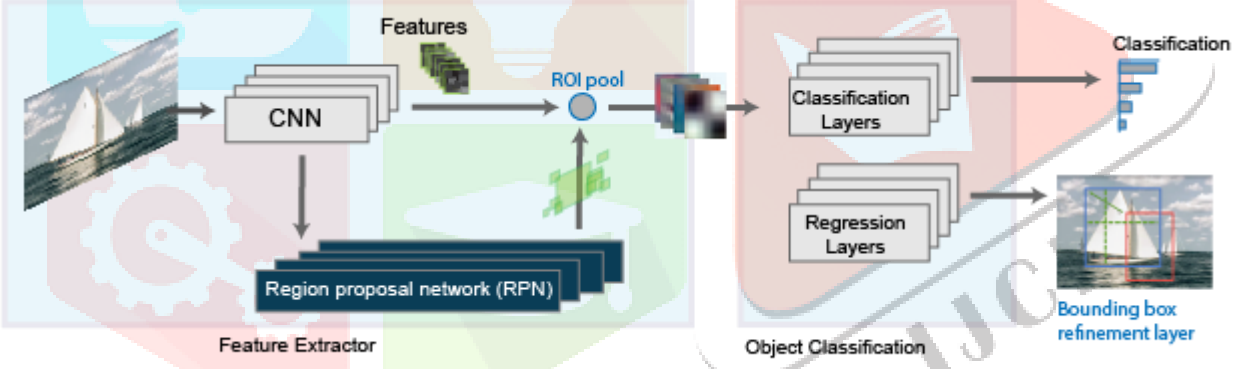


Figure II Faster R-CNN Algorithm's Architecture [17]

A. Faster R-CNN for Fabric Classification: Faster R-CNN consists of two stages; Region Proposal Network (RPN) and Classification Network**.**

a) ***Region Proposal Network (RPN):***

The RPN generates region proposals using an anchor-based approach.

$$L_{RPN}=L_{cls}+\lambda L_{reg}$$ [18]

Where:

L_{RPN}= Total loss of RPN

L_{cls}= Classification loss (object vs. background)

L_{reg} = Regression loss (bounding box refinement)

λ = Scaling factor for balancing classification and regression loss

b) ***Classification Network:***

Once proposals are generated, they are classified into fabric types. The final classification is obtained using:

$$P(C|X) = e^{f(X)} / \sum e^{f(Xi)}$$ [18]

Where:

P(C|X) = Probability of fabric belonging to class CCC

f(X)= Feature vector from CNN backbone

X = Input image

$\sum e^{f(Xi)}$= Softmax denominator ensuring probability distribution

B. CNN-based Feature Extraction:

Feature extraction in CNN is computed using:

$$O = \sigma(W * X + B) \text{ [18]}$$

Where:

O = Output feature map
W = Weight matrix (learned filters)
X = Input image
B = Bias term
σ = Activation function (ReLU, Sigmoid, etc.)

The final classification layer uses Softmax:

$$P(y_i) = e^{zi} / \sum e^{zj} \text{ [18]}$$

Where:

$P(y_i)$ = Probability of fabric type y_i
Z_i = Output from the last layer for class i
$\sum e^{zj}$ = Sum over all class probabilities

V. YOLO

You Only Look Once (YOLO) suggests that an end-to-end neural network be employed that predicts class probabilities and bounding boxes at once. It is different from what has been done by other object detection algorithms, which took existing classifiers and adapted them to make detections. Whereas algorithms such as Faster RCNN operate by identifying potential regions of interest with the Region Proposal Network and then recognizing those regions individually, YOLO makes all of its predictions with the assistance of a single fully connected layer [19].

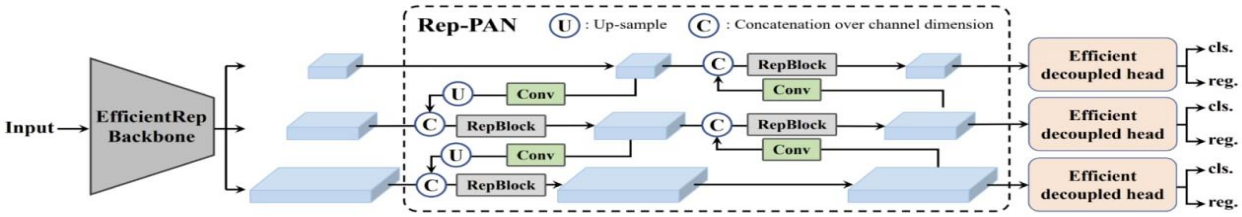


Figure III YOLO Framework [20]

A. YOLO for Fabric Damage Detection:

YOLO makes predictions of bounding boxes, class probabilities, and objectness scores all at once in one forward pass.

$$L_{YOLO} = L_{coord} + L_{conf} + L_{class} \text{ [21]}$$

Where:

L_{coord} = Localization loss (predicted bounding box vs. actual bounding box)
L_{conf} = Confidence loss (measures object presence in a cell)
L_{class} = Classification loss (categorizing object into a class)

Bounding Box Prediction:

$$b^{\wedge} = (x^{\wedge}, y^{\wedge}, w^{\wedge}, h^{\wedge}) \text{ [21]}$$

Where:

$x^{\wedge}$, $y^{\wedge}$ = Center coordinates of the bounding box
$w^{\wedge}$, $h^{\wedge}$ = Bounding box width and height

Intersection-over-Union (IoU) is utilized for comparing predicted bounding box and ground truth bounding box:

$$IoU = A_{intersection} / A_{union} \text{ [22]}$$

Where:

$A_{intersection}$ = Area of overlap between predicted and ground truth box
A_{union} = Combined area of both boxes

VI. METHODOLOGY

Our work, Fashion Material Identification and Its Damage Detection Using Deep Learning, adheres to an organized methodology involving four major phases:

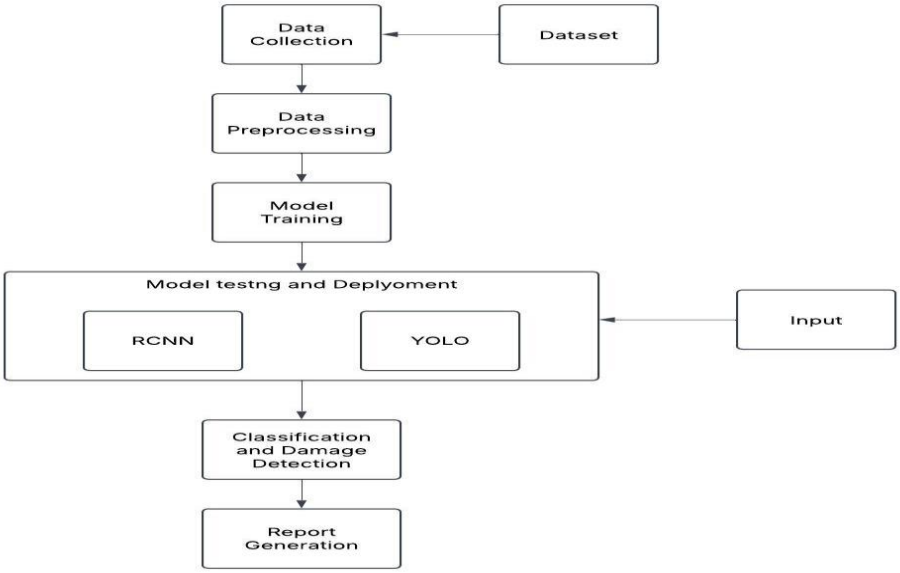


Figure IVArchitecture Diagram

A. Data Collection:

Images were obtained from Kaggle and live camera shots. The dataset contains different types of fabrics (Cotton, Wool, etc.) and types of damage (cuts, stains, holes) at different lighting conditions and angles for increased generalizability.

B. Data Pre-processing:

Images were resized, normalized, and augmented. Label encoding was done for classifying material and damage types. Noise reduction and contrast enhancement were done as well.

C. Model Training and Evaluation:

a) Material Identification: Deployed with RCNN for classifying material types.
b) Damage Detection: YOLO model deployed for real-time defect detection.

Both the models were optimized with Adam and measured using Accuracy, Confidence, Precision, Recall, F1-Score.

D. Model Deployment:

The system was implemented as a Flask web application with image upload and live video support. The frontend was implemented using ReactJS. Predictions are shown in real-time, and PDF reports are created.

This approach guarantees a stable, scalable, and real-time solution for fabric inspection in the textile industry.

VII. RESULTS AND DISCUSSION

Results

Class 1 - hole detection model performance

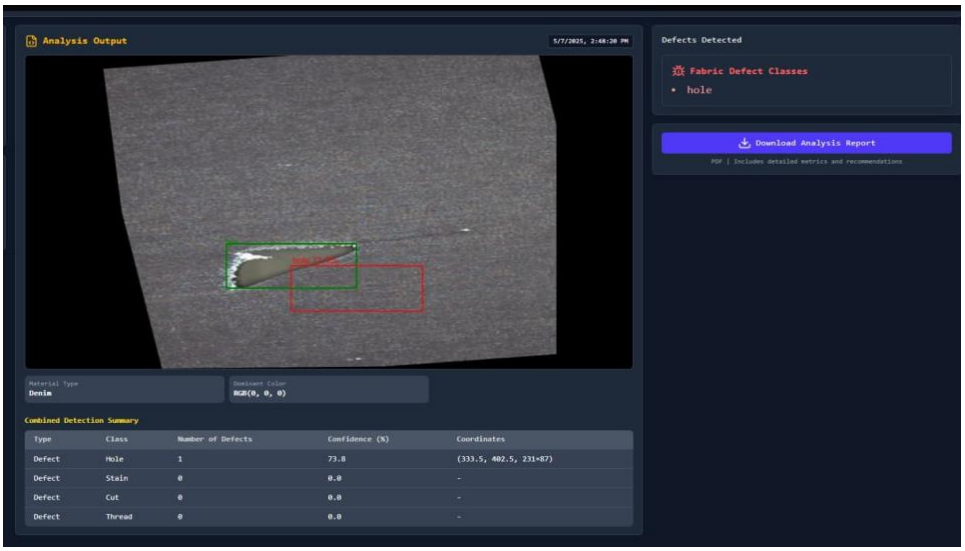


Figure V.Hole detection model performance

The system accurately identified holes in denim fabric with a confidence of 73.8% highlighting the defects using bounding boxes. In one instance, the model also detected a cut in the same fabric sample, showing dual defect detection capability. The RGB values captured indicate color tones close to white and dark gray, validating fabric characteristics. The precise coordinates help in locating the defects visually.

Class 2 - Stain detection model performance

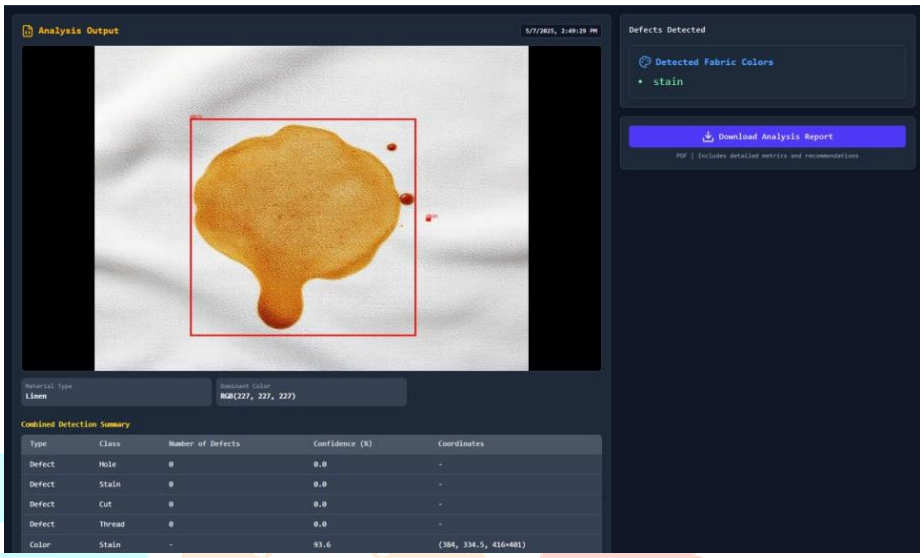


Figure VI. Stain detection model performance

Stains were effectively detected on both cotton and linen fabrics. The cotton sample exhibited a stain with a confidence score of 92.5%, while the linen fabric's stain detection achieved a higher confidence of 93.6%. This high precision confirms the model's reliability in recognizing stains across varied materials and colors, such as off-white tones as indicated by RGB values.

Class 3 - thread detection model performance

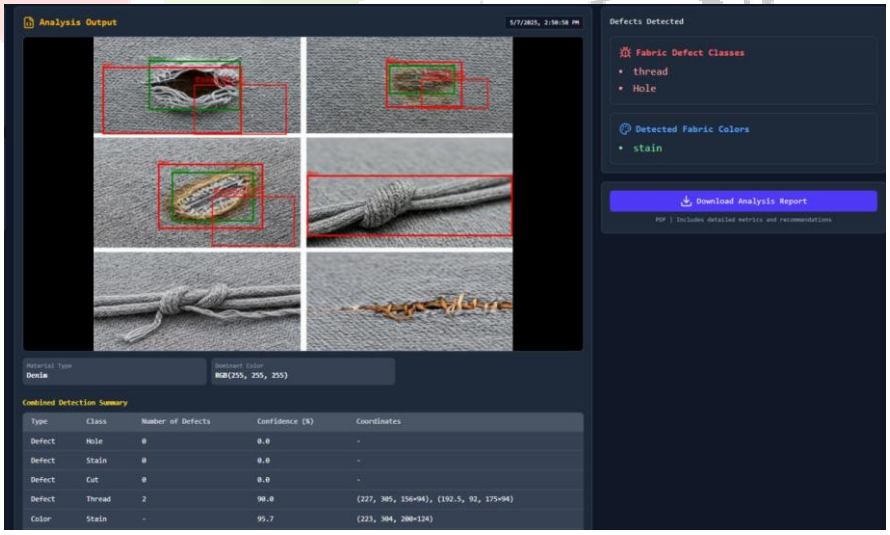


Figure VII. Thread detection model performance

Thread defects were identified on cotton and denim fabrics. The cotton sample showed two types of defects: stain and thread, with confidence levels of 88.4% and 81.1%. The detection was precise, covering multiple bounding boxes indicating thread damage. The denim fabric sample also exhibited thread defects, reinforcing the model's ability to generalize across textures.

Class 4 – cut detection model performance

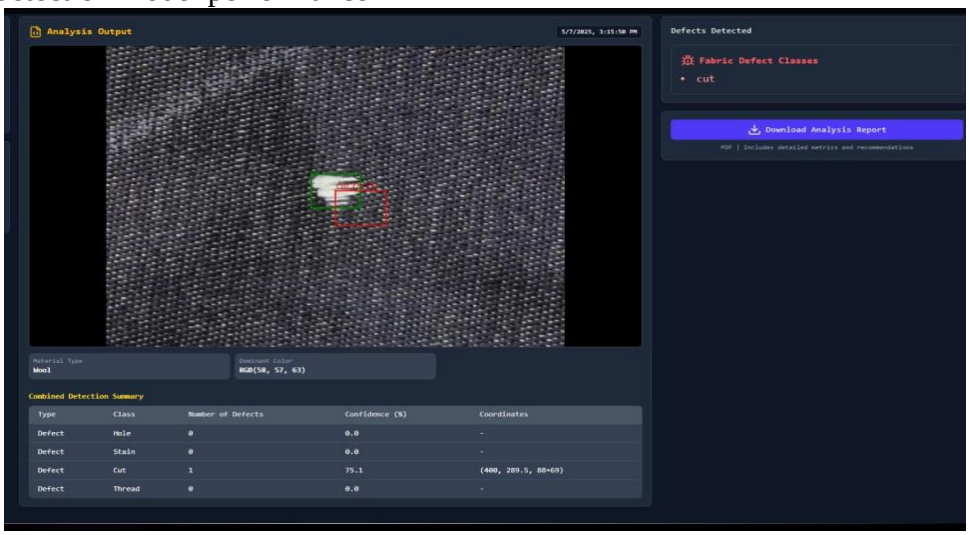


Figure VIII. cut detection model performance

Cuts were detected on denim and wool materials. For denim, both stain and cut were recognized in the same region, showcasing multi-defect detection efficiency. The wool sample had a clear cut with a confidence of 75.1%, and the RGB colour tone confirmed material authenticity. Bounding box coordinates ensure accurate defect localization.

VIII. CONCLUSION

This research presents a deep learning-based system for fabric material identification and damage detection using RCNN and YOLO models. Through comparative analysis, we found that YOLO excels in real-time performance due to its high speed and minimal training time, making it suitable for industrial-scale defect detection. Conversely, RCNN demonstrated superior accuracy and localization precision, which is essential for detailed fabric classification. By leveraging the strengths of both models, our system achieves a balanced trade-off between detection speed and classification quality. The results validate the effectiveness of deep learning in automating fabric inspection, reducing human effort, and improving defect detection accuracy. This integrated approach can significantly benefit the textile industry by ensuring higher quality control standards and operational efficiency.

IX. FUTURE WORK

In the future, the system can be expanded to detect a wider variety of fabric types, like silk, chiffon, denim, and synthetic blends. The system can be made more advanced to detect every kind of fabric defect—like holes, pilling, slubs, and loose threads—with greater accuracy. Also, including detection of printed patterns, embroidered patterns, and woven textures will make the model more versatile. Real-time integration with industrial cameras and automated feedback mechanisms can enable the solution to be scalable to textile manufacturing units. Utilizing larger and more varied datasets will further enhance model robustness and generalization.

X. REFERENCES

[1] https://www.kaggle.com/datasets/rmshashi/abric-defect-dataset.
[2] M. B. Mohammed J. Yousif, “Enhancing the Accuracy of Image Classification using Deep Learning and Preprocessing Methods,” vol. 3, no. 4, pp. 269-281, 2024.
[3] Y. Kim, S. Kim, M. Jeon, “NBBOX: Noisy Bounding Box Improves Remote Sensing Object Detection”, 2025
[4] Sami Naqui, Mohammed EI-Sharkawy, Niranjan Ravi, “BIoU: An Improved Bounding Box Regression for Object Detection,” vol. 12, no. 4, 2022.
[5] Mustafa Dogan, Mehmet Erdogan, “Enhanced Curvature-Based Fabric Defect Detection: A Experimental Study with Gabor Transform and Deep Learning,” vol. 17, no. 11, 2024.
[6] Sanpeng D, Hongchang S, Yuming Q, Ruijun Ma, “An Algorithm for Fabric Defect Detection Based on Adaptive Canny Operator,” pp. 475-481, 2019.
[7] Jingan W, Weidong G, Xiang Jun, “Fabric defect detection based on a deep convolutional neural network using a two-stage strategy,” vol. 91, no. 1-2, 2020.

[8] Christian S, Sergey Ioffe, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift" in International Conference on Machine Learning, Lille, France, 2015.
[9] Rafia M, Muneer A, Arshad A, Mariam Nasim, "'Fabric Defect Detection in Real World Manufacturing Using Deep Learning'," vol. 15, no. 8, 2024.
[10] Zesen Wu, Huosheng Xie, "A Robust Fabric Defect Detection Method Based on Improved RefineDet," vol. 20, no. 15, 2020.
[11] Saima Saleem, "Improving CNN Performance in Fabric Defect Detection with Sampling Techniques," 2024.
[12] Zhiyong Z., Peima W., Kang Gui, "Fabric Defect Detection Based on Cascade Faster R-CNN," Article No.: 98, pp. 1-6, 2020.
[13] Min Z., Junfeng J., Dong Zhuo, "Fabric defect detection using the improved YOLOv3 model," Journal of Engineered Fibers and Fabrics, 2020.
[14] R. Kalaivani, G. Revathy, "Fabric defect detection and classification via deep learning-based improved Mask RCNN," 2023.
[15] C. Wang, Y. Li, M. Gao, J. Li, Qiang Liu, "A Fabric Defect Detection Method Based on Deep Learning," 2022.
[16] A. Lee, M. Hong, Makara Mao, "Efficient Fabric Classification and Object Detection Using YOLOv10," vol. 13, no. 19, 2024.
[17] https://www.mathworks.com/help/vision/ug/getting-started-with-r-cnn-fast-r-cnn-and-faster-r-cnn.html.
[18] K. He, R. Girshick, J. Sun, Shaoqing Ren, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," 2016.
[19] https://www.v7labs.com/blog/yolo-object-detection.
[20] https://www.v7labs.com/blog/yolo-object-detection.
[21] S. Divvala, R. Girshick, A. Farhadi, Joseph Redmon, "You Only Look Once: Unified, Real-Time Object Detection," 2016.
[22] A. F. Joseph Redmon, "YOLOv3: An Incremental Improvement," 2018.