Engineering Science and Technology, an International Journal 53 (2024) 101681

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

journal homepage: www.elsevier.com/locate/jestch

Full length article

Fabric surface defect classification and systematic analysis using a cuckoo search optimized deep residual network

Hiren Mewada [a,*], Ivan Miguel Pires [b,*], Pinalkumar Engineer [c], Amit V. Patel [d]

[a] Electrical Engineering Department, Prince Mohammad bin Fahd University, P.O. Box 1664, Al Khobar 31952, Saudi Arabia
[b] Instituto de Telecomunicações, Escola Superior de Tecnologia e Gestão de Agueda, Universidade de Aveiro, Águeda, Portugal
[c] Department of Electronics Engineering, Sardar Vallabhbhai National Institute of Technology (SVNIT), Surat, Gujarat, India
[d] Wolfson School of Mechanical, Electrical and Manufacturing Engineering, Loughborough University, LE11 3TU, United Kingdom

ARTICLE INFO

Keywords:
Fabric defect
Deep Learning
Industrial Growth
Sustainability labeling
Optimization
Decision-making

ABSTRACT

Fabric defects can significantly impact the quality of a textile product. By analyzing the types and frequencies of defects, manufacturers can identify process inefficiencies, equipment malfunctions, or operator errors. Although deep learning networks are accurate in classification applications, some defects may be subtle and difficult to detect, while others may have complex patterns or occlusions. CNNs may struggle to capture a wide range of defect variations and generalize well to unseen defects. Discriminating between genuine defects and benign variations requires sophisticated feature extraction and modeling techniques. This paper proposes a residual network-based CNN model to enhance the classification of fabric defects. A pretrained residual network, ResNet50, is fine-tuned to classify fabric defects into four categories: holes, objects, oil spots, and thread errors on the fabric surface. The fine-tuned network is further optimized via cuckoo search optimization using classification error as a fitness function. The network is systematically analyzed at different layers, and the investigation of classification results are reported using a confusion matrix and classification accuracy for each class. The experimental results confirm that the proposed model achieved superior performance with 95.36% accuracy and a 95.35% F1 score for multiclass classification. In addition, the proposed model achieved higher accuracy with similar or fewer trainable parameters than traditional deep CNN networks.

1. Introduction

The textile industry has large-scale production with complicated processes. A fault on the surface of a manufactured cloth is referred to as a fabric defect. There are many fabric flaws, most of which result from manufacturing or equipment errors. In addition, defective yarns or machine spoilage might result in flaws. Each aspect has a unique impact and significantly decreases the fabrics' sales and usability. Fabric defect classification is crucial for maintaining product quality, reducing costs, ensuring consumer safety, improving driving processes, meeting regulatory requirements, and facilitating effective supplier management. Fabric defects can significantly impact the quality of a textile product. Defective fabrics can result in waste and increased production costs. The faults in the earlier stage of the fabric production process hurt the later stage. Fabric defect classification provides valuable insights into the manufacturing process. Manufacturers can identify root causes by accurately classifying defects and taking appropriate corrective actions. This can lead to reduced material waste, improved production efficiency, and cost savings. Identifying the defects can ensure that only high-quality fabrics are used in the products. Therefore, early investigation of fabrication faults could reduce the losses suffered by businesses.

The impact of deep learning has considerably exceeded the expectations in industrial automation [1]. Deep learning works on images and signals to reduce industry difficulties and improve the manufacturing process. Fabric flaw identification has historically been performed manually through ineffective and expensive visual inspection. When a flaw is found, the manufacturing process stops, and the specifics of the defect's occurrence and its position are recorded [2]. However, manual processing has adverse effects. This may reduce production efficiency. The operator may miss smaller defects or need extensive effort to localize the defect in the fabric. Currently, deep learning plays an essential role in fabric defect segmentation [3] and classification [4].

The learning rate, regularization parameter, network depth, number of filters, and filter size are among the hyper-parameters commonly used in deep learning networks. These parameters are typically obtained through experience and are defined relative to broad terms;

* Corresponding authors.
E-mail addresses: hmewada@pmu.edu.sa (H. Mewada), impires@ua.pt (I.M. Pires), pje@eced.svnit.ac.in (P. Engineer), a.patel2@iboro.ac.uk (A.V. Patel).

https://doi.org/10.1016/j.jestch.2024.101681
Received 3 December 2023; Received in revised form 12 March 2024; Accepted 24 March 2024

however, they significantly affect the effectiveness of classification. Finding an optimum value of these hyperparameters is a significant and challenging task. These hyperparameters can be categorized as integer, continuous, or mixed according to their value type assignments. The gradient algorithm is primarily used for continuous-type hyperparameter optimization. Most researchers use evolutionary algorithms based on particle swam [5] and genetic algorithms [6] for hyperparameter optimization. However, nearly no analogous techniques for mixed hyperparameter optimization problems exist, except for Optuna [7], a built-in hyperparameter optimization algorithm in Python that has just begun to be utilized to address deep learning hyperparameter optimization issues. Given that the CS algorithm has demonstrated superior performance to other intelligent evolutionary algorithms and has proven helpful in solving a wide range of optimization issues [8], this research utilizes the cuckoo search method for hyperparameter optimization.

This paper presents a new approach using a deep convolution network. Texture patterns can exhibit intricate and subtle details that are crucial for accurate classification. ResNets excels at capturing fine-grained information due to its deep architecture. By stacking multiple residual blocks, ResNet can learn hierarchical representations of texture patterns, progressively capturing low-level and high-level features. It helps the network understand and discriminate between different texture patterns, even those with subtle variations. Therefore, a new approach using the ResNet architecture is presented in this paper. The ResNet architecture is pretrained using the ImageNet dataset. Thus, the network's learning rate, number of filters, and size are fixed according to the training experience from ImageNet feature sets. The learning rate and drop factors are optimized using the cuckoo search algorithm to fit this network well for our defect classification problem. The proposed network is analyzed in depth, examining the results at different layers. A well-established comparison with various literature demonstrating the strength of the proposed network is presented. The proposed network is analyzed in detail, and the results at different layers are examined. A well-established comparison with various literature reports demonstrating the strength of the proposed network is presented. The contributions of the paper are as follows:

- A high diversity in size, shape, texture, and appearance of defects in the fabric makes it challenging to classify. We presented a deep CNN, ResNet, to classify defects in these challenging environments. A ResNet network providing intricate features and textures to discriminate defects well is presented to classify defects in fabrics with texture patterns.
- A limited data size creates a risk of overfitting, and the network fails to be generalized well. Therefore, we augmented the datasets by presenting autoorientation and flipping images horizontally and vertically. The fabric could be skewed in either direction during the data collection process. Thus, random autoorientation along with vertical and horizontal flipping increases the variation in the position and orientation of the defect over the surface. It combats the overfitting problem of the network.
- A transfer-learning approach is used instead of training a network from scratch. A pretrained ResNet model has learned to extract general features and patterns from useful images across various tasks. Using a pretrained model, one can leverage the knowledge and representations learned from a large dataset and transfer it to fabric defect classification. This approach can save significant computational resources and training time.
- The learning rate, a tuning hyperparameter, determines the step size at each iteration and provides fast convergence if used correctly. Therefore, a cuckoo search optimization is used to optimize the learning rate and decay factor.
- A systematic analysis of the network's outcomes is presented. Initially, the network's performance for each defect is individually presented, and later, the model's overall performance is presented. The study also shows where the model finds ambiguity in the classification. It provides insights into the specific patterns or instances that lead to misclassifications. By examining the misclassified samples, common characteristics, challenging scenarios, or classes that are prone to confusion can be identified. This analysis helps in understanding the model's limitations and weaknesses and guides improvements.
- Finally, a comprehensive comparative analysis with state-of-the-art networks is presented using the accuracy, F1-score, top-10 ambiguity score, and learnable parameters of the network.

Overall, the structure of the paper is as follows: The relevant works are presented in Section 2, and the suggested model is presented in Section 3. The analytical findings are presented in Section 4. Finally, Section 5 summarizes them with feature scopes.

2. Literature review

Most literature findings are focused on whether fabric is defective or not. Therefore, most dataset developed by the researchers was categorized into two classes e.g. defective and non-defective and its annotation by localizing the defect was provided. A limited dataset provides a classification of defect types. The study of these private datasets was presented well in [9]. Table 1 summarizes the dataset available publicly with its characteristics. This study uses optical images presented in [10,11].

The rapid growth in computer vision and image processing algorithms and advancements in machine learning techniques have replaced manual defect detection and classification with autonomous approaches. Upon fabric inspection, numerous faults are discovered, including drop stitches and color shading variance [16,17]. The defect detection techniques can be classified into statistical, structural, model-based, and learning-based techniques [18]. Statistical methods extract features that determine similarity and regularity using the mean and variance from an image plane [19]. The sparse approximation using image patches and its dictionary formation can minimize the features in image classification [20]. A similar approach to fabric flaw detection was presented in [21]. These approaches are ineffective for low-contrast and patterned fabric images. The spectral methods used frequency domain approaches, i.e., Fourier transform, wavelet transform [22,23], local binary pattern [24], and gray-level co-occurrence matrix [25], to obtain spectral features from textured regions. Model-based algorithms, i.e., Gaussian mixture model [26], are not robust and depend primarily on data. Therefore, this paper focuses on machine learning-based algorithms.

An active contour can segment the desired region, and if the edges of the regions are enhanced well, the active contour can provide accurately segmented regions [27]. Bumrungkun [28] extracted edge features using active contour, and subsequently, a support vector machine (SVM) was used to classify five defect types. This method fails for images with poor brightness and contrast. In Ref. [29], an Inception v3 model was used to extract features from texture fabric images. Later, a machine learning model was optimized and trained using these features for the classification and presented an improvement of 6% in classification compared to traditional CNNs. A binary classification approach, i.e., defective vs nondefective fabric, was presented in [30]. They segregated the feature extraction and classification models using convolution features from CNN and SVM as a classifier. They observed that ResNet18 with SVM performed better, with a 94.66% classification rate. Meeradevi et al. [31] presented a review study of 79 research papers on detecting fabric defects and shade variation. They found that the deep learning model performed best among all traditional and machine learning models.

A convolution model is adopted using a separable convolution process in the CNN network; the fabric defect location is identified in [32], and their model achieved 98.01% segmentation accuracy. However, the

Table 1
Publicly available dataset.

Ref	Proprietor	Characteristics
[12]	Intellisense Lab, Department of Computer Science and Engineering, University of Moratuwa	Three classes of defective images i.e. horizontal, and vertical lines and holes with texture background. Prepared for Defect localization Mask provided without annotation of defect types. Images are categorized as defective or non-defective.
[13]	AFID: A Public Fabric Image Database for Defect Detection	247 RGB images with 117 labeled objects belonging to 12 different classes. 106 defective images for 12 different classes. Prepared for Defect localization. 59% of images are without annotation of defect type. A small set of images for each defect type. Images are categorized as defective or non-defective.
[14]	AITEX Fabric Image Database	245 RGB images of 7 different fabrics 140 defect-free images (20 images for each class) and 105 defective images. Labeled images with defects are available. A small set of images for each defect type. Images are categorized as defective or non-defective
[15]	TILDA Textile Texture-Database	61 defects covering images having a uniform texture and complex background texture. 50 images for each defect. RGB images of each defect. A large set of images with annotation of the defect. Primary application is to segment defects on fabric surfaces.
[10]	Roboflow dataset	Images of four defects including holes, oil spots, object and thread errors. Optical images of each defect. A large set of images with appropriate labeling and the same resolution.
[11]	DAGM 2007 competition dataset	a synthetic dataset for defect detection on textured surfaces A weakly labeled dataset Ten types of texture-pattered images with defects and without defects. Binary Labeling is provided for each class without annotation.

type of defect is not classified in their model. Furthermore, a binary classification model using a depthwise separable convolution network was presented by Chen et al. [33]. An Xception network was adopted and tested on their dataset containing defective and nondefective fabrics. However, their model accuracy was limited to 93.57% for binary classification.

Zhao et al. [34] adopted the VGG16 network by adding a squeeze-and-excitation and SSD module. The course features were obtained using the squeeze operation on convolutional features, and later, the weights of layers were determined using the excitation operation. A comparison with MobileNetV3 and EfficientNet was presented for multiclass classification with an average accuracy of 81.7%. Using the transfer learning method, a pre-trained VGG16 network was tuned and features were extracted from fabric surfaces. They reduced the future size using principle component analysis. A CNN network adopted replacing the softmax layer with a moth flame-optimized parallel chaotic search (PCS) algorithm. The author showed that this model detected defective fabric accurately for regular patterned fabrics, unpatterned fabrics and irregularly patterned fabrics [35]. Guo et al. [36] presented a YOLOv5 network with 95% accuracy to detect defects of lines and holes in fabric surfaces using their dataset. Their model performs well if the image of the fabric is clear. Otherwise, it fails to detect and classify the defect. A tiny YOLO network was presented in [37] to use on-edge devices for binary classification, and the model achieved superior results of 99%.

The Cascade R-CNN technique is the foundation of a fabric defect detection dataset in [38], and three tricks are presented to improve the precision of the detection algorithm. Initially, they used a multiscale approach for training the network. Then, dimension clustering was applied to the dataset, and the model was trained. The defect size (i.e., a box of width and height) from the images was adjusted before training in dimension clustering mode. Finally, a nonmaximum method was used to preprocess the dataset, and the model was trained using these new datasets. They performed binary classification, i.e., defective and nondefective fabric classification. The results were evaluated using the intersection of union parameters. Another cascaded approach integrating a faster R-CNN network with a feature pyramid network was presented in [39]. The feature pyramid network allows shallow features with deep features of R-CNN, improving the semantic segmentation of defect location with 95% precision. In addition, the type of defect was also retrieved from the segmented region, but the classification accuracy was not investigated.

A lightweight CNN can easily fit to mobile platforms and allows its use in resource-constrained devices [40]. A lightweight CNN requiring fewer learnable parameters was presented and compared with VGG16, MobileNetV2, and EfficientNet to classify fabric surface images as either defective or not [41]. They validated the model with their datasets and presented a 99% F1-score. State-of-the-art comparisons are limited to standard CNNs. A similar fabric defect detection model using the CNN network was presented by Huang and Xiang [42], where the CNN model was used to obtain features of repetitive patterns of the fabric, and later semantic segmentation was used using the Ghost network. An IoU measurement of 77% was presented for detecting fabric defects using the Tilda dataset. Liu et al. [43] also proposed a binary classification approach by optimizing the VGG16 neural network. However, the network is manually optimized by visualizing the features. Lin et al. [44] focused on defect classification for multicolor fabric. They preprocessed images using various filtering methods, a CNN transfer-learning approach was used, and the model was trained to detect

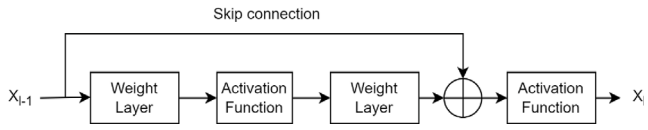

Fig. 1. Structure of residual layer in the ResNet architecture.

defects. However, they validated the model with limited raw data and without external data.

In Ref. [45], the authors checked the performances of seven feature sets and three machine learning-based classifiers, namely, KNN, SVM, and decision tree on the Tilda dataset. A total of six defects were classified using these models. Overall, KNN performed better, with an average accuracy of 90.2%. A defect detection and classification model using two stages was presented in [46]. Initially, the U-Net network was used to determine whether defects existed in the images of the AITEX dataset. Later, the VGG16 network and random forest algorithm were used to classify defects in the defective fabric. Although the model detection accuracy is high, the classification rate is relatively low, and the model fails to classify nep and cut salvage defects.

The study showed that deep CNNs are widely used in fabric defect identification, and most studies have detected the presence of fabric defects and their location, irrespective of the type of defect. Among all networks, the YOLO network achieved superior accuracy in detecting defective fabrics. A limited number of studies have presented classifications among the types of defects. However, the lack of detailed results analysis of these models is a common weakness for these models.

3. Methods

3.1. Overall structure of the proposed model

Deep learning-based models are frequently used for image classification tasks and have exhibited their efficacy by producing outstanding results in many varied use cases. The main challenge in fabric defect classification is its background. In the presence of a textured background, global as well as, local features also participate in the classification. But traditional CNNs use global features only and therefore, they might underperform in fabric defect classification. This section develops a deep CNN model that can automatically identify subtle defects in fabric from fabric surface images. A defect in the fabric can be treated as a violation of patterns on the surface. Detecting violations in texture patterns can be effectively achieved by leveraging fine-grained details. Texture patterns often exhibit specific regularities and structures. Any deviation or violation from these patterns can indicate a defect or anomaly. Fine-grained details enable the detection of subtle variations, irregularities, or inconsistencies within the texture pattern. By analyzing intricate elements and small-scale features, such as texture gradients, local textures, or pixel-level variations, it becomes possible to identify deviations that may not be apparent at a coarser level of analysis. Fine-grained details provide a level of granularity that allows for the precise localization and characterization of texture pattern violations, facilitating the detection of defects or abnormalities on the fabric surface.

The proposed model uses a residual network for subtle defect classification from the fabric surface. ResNet's key innovation lies in introducing residual connections, which allow the network to learn residual mappings. By incorporating skip connections that bypass specific layers, ResNet enables the direct flow of gradients to earlier layers during training, effectively addressing the vanishing gradient problem. Fig. 1 shows the residual layer connection where the lower branch processes the features from the previous layer and the upper branch passes the features without any processing. The subtraction of the lower and upper branch features provides the residual features, i.e., fine-grained details.

A function $F()$ can be expressed using the input features x and weights W in layers. Therefore, the output of the ResNet structure is expressed as

$$y = F(x, W_i) + x \tag{1}$$

In this equation,

- x represents input to the residual layer,
- $F(x, W_i)$ denotes the residual mapping, which is a sequence of convolutional layers with weights W_i.
- The output of the residual block is represented by y.
- The "+" operation indicates elementwise addition.

The residual mapping $F(x, W_i)$ can be further decomposed as follows:

$$F(x, W_i) = W_2 * \sigma(W_1 * x) \tag{2}$$

In this decomposition:

- σ denotes a nonlinear activation function, such as ReLU.
- W_1 and W_2 are learnable weights associated with the convolutional layers.

Overall, the ResNet architecture consists of stacking multiple residual blocks together. Each residual block performs a residual mapping $F(x, W_i)$ and adds the input x to the mapping output, resulting in the transformed output y. This formulation allows the network to learn the residual information effectively and facilitates the training of deeper neural networks.

This architectural design empowers ResNet to capture intricate and subtle details in the data, making it particularly well-suited for tasks that require fine-grained analysis, such as texture classification and object detection. The residual connections enable the network to focus on learning the incremental changes or deviations from the input data, leading to more efficient parameter optimization and improved performance in capturing fine details. The depth of ResNet, achieved through stacking multiple residual blocks, allows for the hierarchical learning of features, enabling the network to extract and exploit both low-level and high-level information progressively. The depth of ResNet plays a crucial role in its ability to extract both low-level and high-level information. By stacking multiple residual blocks, ResNet creates a deep architecture that captures increasingly complex patterns and features. As information flows through the layers of a deep ResNet, each layer learns to extract and represent specific features at different levels of abstraction. The initial layers capture low-level details such as edges, corners, and textures, while the subsequent layers build upon these representations to capture more abstract and high-level features. This hierarchical learning enables the network to understand the data at multiple scales and levels of complexity.

ResNet's skip or residual connections facilitate the smooth propagation of gradients throughout the network, allowing for adequate information flow and gradient updates during training. It enables deep ResNet models to be trained successfully and prevents the degradation problem that can occur with very deep networks. By leveraging the depth of ResNet, the network can capture and combine features from different layers, allowing for the integration of low-level and high-level information. This ability to extract and combine features from various depths enables ResNet to capture fine-grained details while also capturing global context and semantic information. The deep architecture of ResNet thus contributes to its capacity to learn and represent a wide range of features, making it highly effective in tasks that require extracting both low-level and high-level information, such as image recognition, object detection, and semantic segmentation.

Fig. 2 shows the flowchart of the proposed system. ResNet50 has deeper layers, requires less training time than does the ResNet34 architecture, and has significantly fewer trainable parameters than do the ResNet101 and ResNet152 networks. Therefore, a pretrained ReseNet50 architecture, which is trained using the ImageNet dataset, is used, and this network is retrained using the fabric defect dataset.

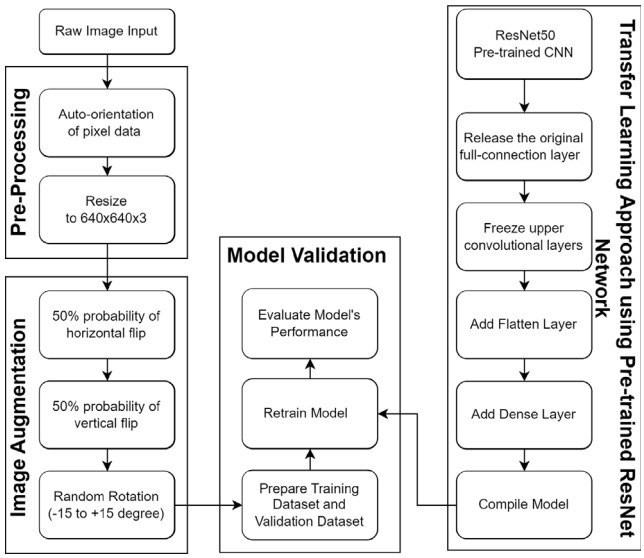

Fig. 2. Flowchart of the use of the pretrained ResNet50 network for defect classification.

Initially, the weight parameters of the pretrained ResNet50 network are obtained to release the original fully-connected layer. Then, the fully connected layer is replaced with a layer designed for four-class classification. Once the model is compiled successfully, the model is re-trained using a fabric dataset to tune the model for defect classification. A data augmentation and k-fold classification approach is used, which suppresses the overfitting problem of the network. The experimental results obtained using this network are discussed in the next section.

3.2. Hyperparameter optimization using the cuckoo search algorithm

Hyperparameters and parameters significantly impact the performance of a CNN. Backpropagation algorithms are mostly used to optimize weights and biases of CNN networks with an objective function of minimizing a loss function. They may suffer from issues like getting stuck in local optima or the curse of dimensionality. In such cases, metaheuristic algorithms like Cuckoo Search provide an alternative by exploring the search space in a non-gradient-based manner, potentially overcoming these limitations. The cuckoo search efficiently explores large search space and therefore, it can find optimal or near-optimal combinations of hyperparameters in comparison to gradient-based methods. By applying cuckoo search (CS), we can discover novel ResNet architecture that may have superior performance or efficiency compared to manually designed architectures.

The CS algorithm emulates the search behavior of cuckoos in nature as they seek the best nest for incubating their eggs. The cuckoo in the algorithm selects nests of better quality to nurture its eggs and enhances nests of lower quality. The key feature of this optimization process is that the cuckoo employs the Levy flying mode and continuously refers to the optimum nests discovered thus far to create a very efficient optimization strategy. We used the cuckoo search algorithm to find the optimum value of the learning rate and its drop factor.

The learning rate determines the step size at which the optimization algorithm updates the neural network weights during training. A high learning rate can cause the optimization process to converge quickly but may overshoot the optimal solution. On the other hand, a low learning rate can help the optimization process converge more accurately but may take a more time. The learning rate schedule adjusts the learning rate over time to strike a balance between convergence speed and accuracy. The learning rate typically decreases gradually as training progresses. The code implements the learning rate schedule using a piecewise function. Let α represent the initial learning rate. It determines the starting point of the learning rate schedule. β represents the learning rate drop factor. It determines how the learning rate is reduced during the training process. The learning rate is multiplied by beta every LearnRateDrop Period epoch. We can find the best values that minimize the classification error on the test set by optimizing these α and β parameters during the cuckoo search optimization. The pseudocode to optimize α and β of the network using the cuckoo search algorithm is presented in Algorithm-1.

Algorithm 1 Cuckoo search based hyperparameter optimization of ResNet

```
Define range of α and β
Initialize fitness function F_i = classification_error(images, classes, α, β)
Generate initial n host nests
Evaluate fitness function F_i
Find the best nest and best fitness function F_i
while t < Max_iteration do
    Generate new host nests using Levy flight
    Evaluate fitness function F_j using new nests
    if F_j > F_i then
        Update best nests and best fitness function F_i
    end if
    Randomly replace some nests with new random solutions
end while
Post process the results
```

3.3. Feature extraction

The ResNet-50 network comprises an input layer, five stages of 49 convolutional layers, and an output layer. The five stages contain one convolutional layer and four residual layers, as shown in Fig. 3. As seen in Fig. 3, each stage comprises multiple residual blocks of varying sizes and widths. Stage 1 extracts a feature map from the input image using a 7×7 convolutional filter, and this stage employs 64 convolutional filters, each of which executes a convolutional computation on the input image. The stride value is set to 2, the interval between which the convolution filters are applied to the input image without overlapping. Thus, the filter moves by two pixels, and the size of the output feature map is calculated by dividing the size of the input picture by the stride value. This stage generates 64 feature maps.

Stage 2 has a residual block with the size shown in Fig. 3. The downsampling of the image by 4 in this stage generates 256 feature maps. This downsampling process allows large-size filters in subsequent stages, improving the network's performance by allowing it to process more data. The first layer in this segment has a convolutional layer of size 1×1 that increases the number of channels of the feature map size from 64 to 256. To maintain a constant channel number, a 3×3 convolutional layer is employed. Furthermore, the 1×1 convolutional layer expands the size of the feature map to 256. The presence of the residual layer in this stage extracts the fine-grained information using skip connections and features from the previous layer.

Stage 3 reduces the size by 1/8. Thus, a 1×1 convolution is employed initially to limit the number of channels of the input data to 128. The feature map is then processed with a 3×3 convolution, reducing the number of output channels back to 128. The output channels are then increased to 512 using a 1×1 convolution. It reduces the size of the input data by 1/8, resulting in a total of 128 feature maps. Stage 3 shrinks the input data while expanding the feature map size. As a result, the model can extract more diverse and complex information from the input image.

Stage 4 has three residual layers. The first layer of the 1×1 convolutional operator reduces the number of channels in the input feature maps. Again, the channel numbers increase using the remaining 3×3

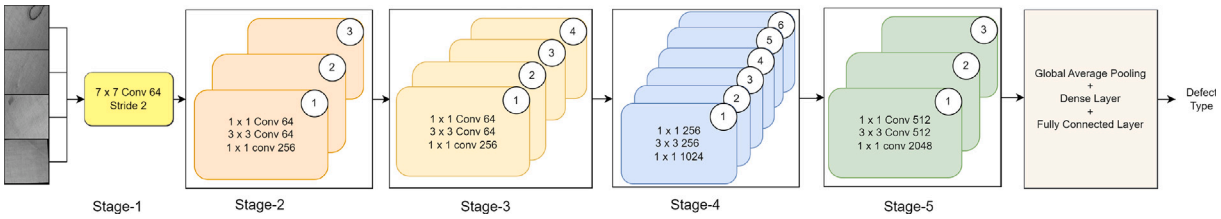

Fig. 3. ResNet50 for feature extraction.

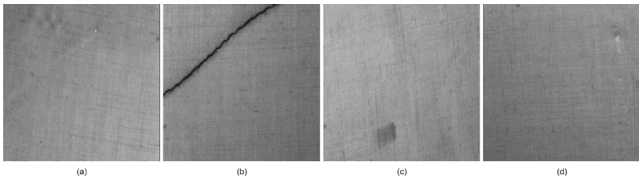

Fig. 4. Sample images of fabric defects dataset (a) hole, (b) object, (c) oil spot, and (d) thread error in fabric.

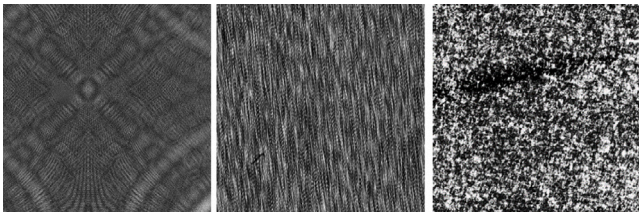

Fig. 5. Sample images of three defects from DGAM dataset.

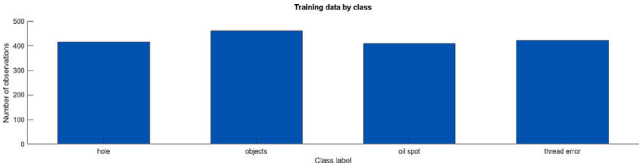

Fig. 6. Training images (70%) used in the experiment from dataset-1.

Fig. 7. Validation images (30%) used in the experiment from dataset-1.

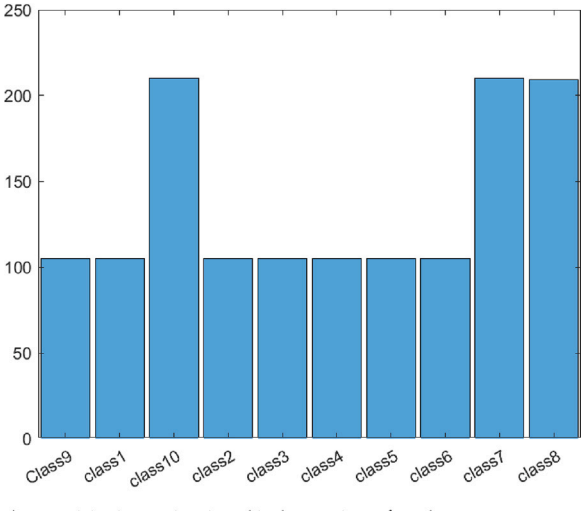

Fig. 8. Training images (70%) used in the experiment from dataset-2.

and 1 × 1 convolution operations. The final output is obtained using a residual connection, which adds the output feature map to the input feature map. Various features are taken from the input feature map throughout this procedure, resulting in a more sophisticated feature map. The feature map created in stage 4 is 14 × 14 in size and has 1024 channels. Downsampling converts the 14 × 14 feature map created in stage 4 to a 7 × 7 feature map. Using two convolution layers and max pooling, the size of the feature map is halved with a stride of 2 × 2. The generated 2048-channel feature map is then used to create a one-dimensional vector by calculating the average value for each channel via average pooling. Pooling is used to decrease the number of optimization parameters. Similarly, the feature map is processed using the convolution operation in stage 5. Finally, feature maps are passed through average pooling, dense layers, and fully connected layers, generating the probability of class prediction.

4. Experiment results and analysis

The Resenet model is used and retrained for the two datasets. The first fabric dataset from Roboflow [10] is used to test the proposed model. This dataset includes 2441 images for all types of defects, including holes, objects, oil spots, and thread errors. Fig. 4 shows a sample image of surface defects from the dataset. The second dataset DAGM 2007 [11] providing ten defects each with a different textured background is used. The dataset-2 contains 1950 images with 150 images for 8 classes and 300 images for remaining two classes. The sample images of three classes are shown in Fig. 5.

From the total images, 70% of the images are used to train the model, and 30% of the images are reserved for testing the model. Figs. 6–9 visualize the images used to train and validate the models during the experiments.

A considerable discrepancy in the dataset's image size and total number of images will result in a data imbalance problem, leading to model overfitting. Simultaneously, if it is used to train hardware constraints, the initial image resolution will increase, and the computational burden during training will increase. As a result, before model training, the training data must be preprocessed. Therefore, the following preprocessing is applied to each image:

- Auto-orientation of the pixel data (with EXIF-orientation stripping)
- Resize to 640 × 640

Furthermore, to increase the size of the images during the training process, the following augmentation was applied to create 3 versions of each source image:

- 50% probability of horizontal flip

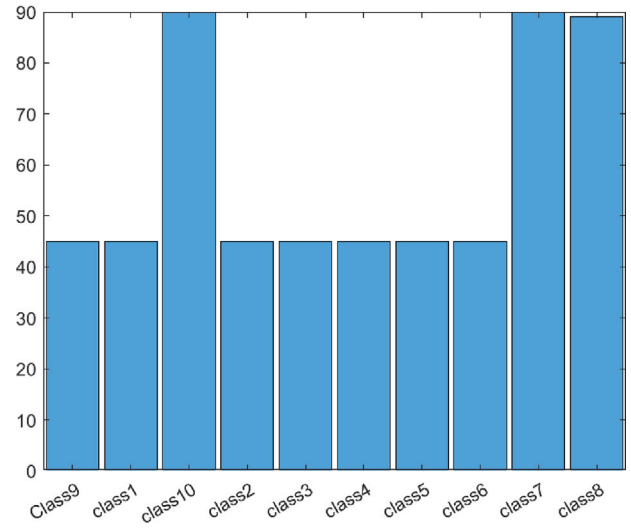

Fig. 9. Validation images (30%) used in the experiment from dataset-2.

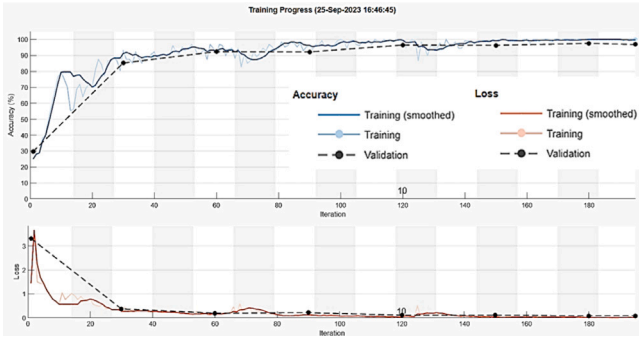

Fig. 10. Accuracy and loss variation over the epochs for the train and validation subsets from dataset-1.

- 50% probability of vertical flip
- Random rotation between −15 and +15 degrees

It is necessary to adjust the network's hyperparameters to prevent overfitting during training to improve the model's classification rate. In a convolutional network, the top layers extract more general features, such as outlines or shapes, while the bottom layers extract more specific features, such as edges, textures, and colors. In this study, the top layers of the network were kept and retrained to retain course features relevant to defect features. The training and validation accuracy and loss were monitored for each epoch to prevent overfitting problems. A decrease in the accuracy of the validation data indicates the occurrence of overfitting. Therefore, an early stop and data augmentation are adopted to prevent this issue. The network training will be stopped if the validation accuracy is not optimized after 15 epochs of training.

To optimize the two hyperparameters, the learning rate and its drop factors were initialized to [0.001, 0.9], and the intervals for these parameters were $\alpha = [0.0001, 0.01]$ and $\beta = [0.1, 0.99]$. The number of filters and filter size are kept constant from the pretrained model. Figs. 10 and 11 shows the accuracy and loss over the epoch for both the training and validation sets. The model achieved 100% accuracy for the train subset and 95.36% accuracy for the validation subset for dataset-1. For dataset-2, model received 100% accuracy.

Fig. 12 displays confusion matrices that validates the model's performance. It provides insights into which classes were predicted successfully and flags any potential faults with the classifier. The results show that the model effectively predicted the defects, with the lowest

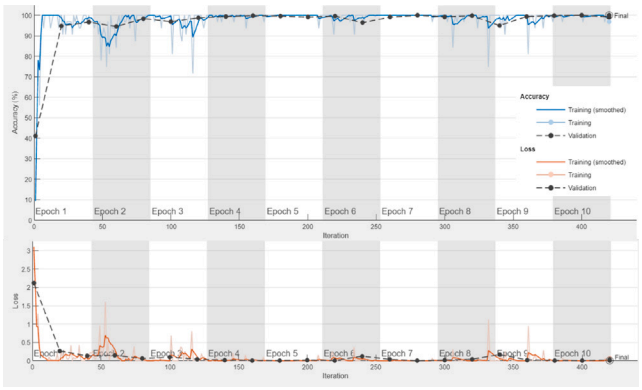

Fig. 11. Accuracy and loss variation over the epochs for train and validation subsets from dataset-2.

Table 2
Top 10 classes with the most ambiguity.

Image	Ambiguity	Likeliest	Second	True Class
331	0.996526	'objects'	'oil spot'	'objects'
249	0.994408	'thread error'	'objects'	'objects'
527	0.981387	'oil spot'	'objects'	'oil spot'
477	0.966847	'oil spot'	'objects'	'oil spot'
474	0.96532	'oil spot'	'objects'	'oil spot'
529	0.960491	'oil spot'	'objects'	'oil spot'
368	0.953623	'oil spot'	'objects'	'objects'
281	0.937232	'objects'	'oil spot'	'objects'
87	0.935936	'thread error'	'hole'	'hole'
550	0.932402	'objects'	'oil spot'	'oil spot'

classification rate for fabric defects compared to that for other defects for dataset-1. However, 12 images were misclassified as another defect type. Fig. 13 showcases a few sample images that were classified negatively by the network. These images are complex to classify, and even manual classification can lead to errors in prediction. The receiver operating characteristic (RoC) plot displayed in Fig. 14 helps analyze the model's overall response. The RoC curve illustrates the trade-off between the true positive rate (sensitivity) and the false positive rate (1 - specificity) at various classification thresholds. The RoC plot covers a 96% area under the curve, indicating that the model's performance is better.

For dataset-2, model received 100% accuracy. The dataset-2 has variation in the textured background and therefore, it identifies the pattern well. The main limitation of this dataset is that it does not contain defect variation with a similar textured background which makes it challenging for classification.

The model is analyzed in detail for dataset-1, observing the ten classes those are most difficult for the model to classify accurately. These classes represent the areas where the model exhibited the highest level of uncertainty or confusion. Table 2 shows the model's ambiguity, in which it observed the oil spot defect but likely with a defect of the object type. The model predicted well, but the thread error defect was misclassified as an object. The quantitative analysis of the model was performed by calculating its classification accuracy, precision, specificity, and F1 score for each defect, as well as overall. Tables 3 and 4 present the detailed results of this analysis.

To understand more about network prediction for dataset-1 and to check that the proposed network focuses on the right region within image to identify the type of defect, a gradient-weighted class activation mapping (Grad-CAM) map of images was plotted. Grad-CAM provides a heatmap-like visualization that highlights the important regions in the input image. Fig. 15 plots the result over the original image with transparency to see which areas of the image contribute most to the classification score.

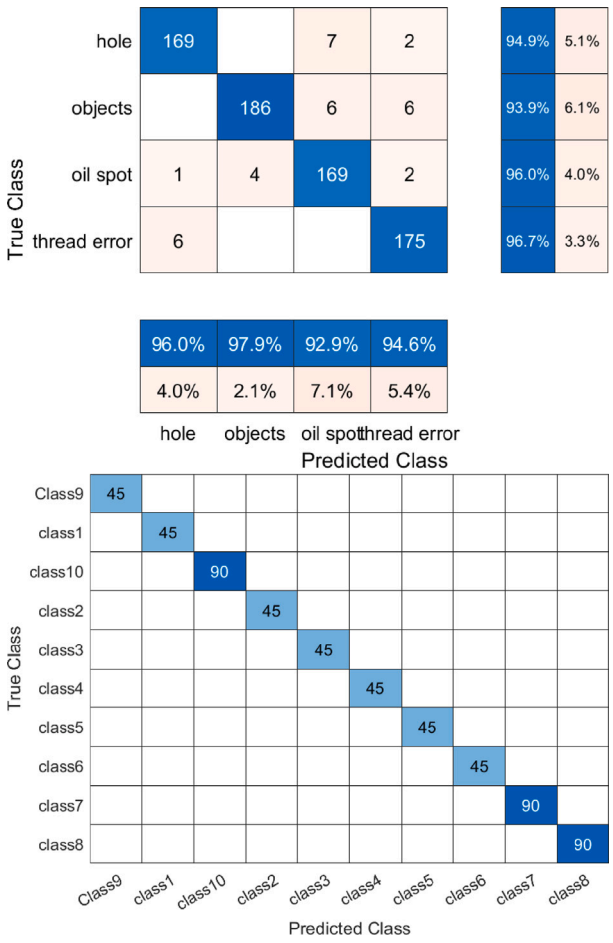

Fig. 12. Confusion matrix for dataset-1 and dataset-2.

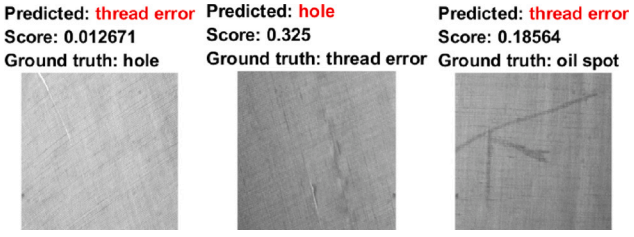

Fig. 13. Sample images where the model fails to predict.

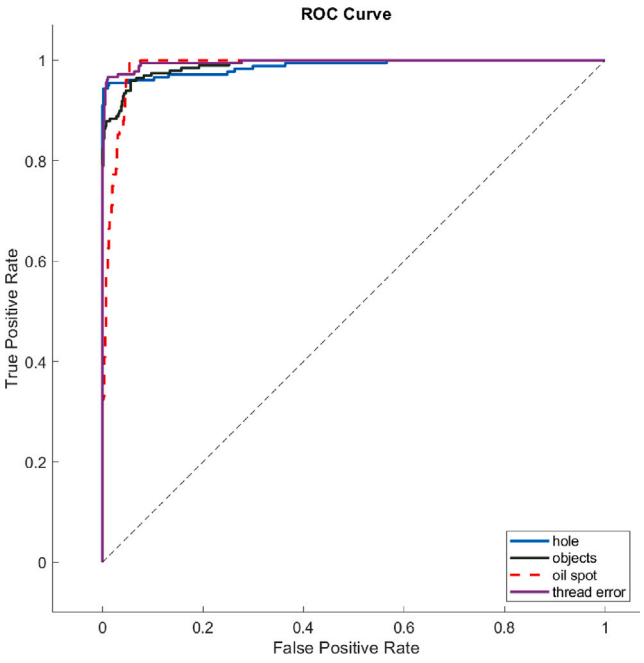

Fig. 14. ROC curve .

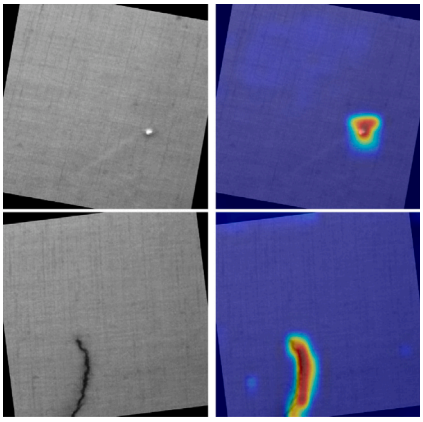

Fig. 15. Grad-CAM for sample images (left) original image (right) Grad-CAM of image showing important region.

Accuracy is a widely used measure that evaluates the overall correctness of a classifier's predictions. It computes the proportion of accurate predictions (true positives (TP) and true negatives (TN)) to the total number of instances (N) in the dataset. The accuracy is expressed as a percentage, where higher values indicate better performance.

$$Accuracy = \frac{TP + TN}{N} \tag{3}$$

Precision is calculated by dividing the total number of true positives by the total number of instances predicted as positive (including both true and false positives). In simpler terms, precision provides insights into how effectively the classifier can avoid false positive errors.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

When evaluating a classifier's performance, specificity measures its ability to identify negative instances correctly. It is calculated by dividing the number of true negatives by the total number of instances predicted as negative (true negatives plus false positives). Specificity complements precision by focusing on the true negative rate. A higher specificity indicates a lower rate of false negatives, meaning that the classifier can accurately identify negative instances. In simpler terms, the specificity metric tells us how well a classifier can identify instances that do not belong to a specific class, helping us to ensure that negative cases are not falsely labeled as positive.

$$Specificity = \frac{TN}{TN + FP} \tag{5}$$

The F1 score is a harmonic mean of precision and recall (also known as the sensitivity or true positive rate). It provides a balanced measure of the classifier's performance, considering false positives and negatives.

$$F1 = \frac{2X\,Precision X\,Recall}{Precision + Recall} \tag{6}$$

Table 3
Prediction results analysis for each defect individually from dataset-1.

Parameter	Accuracy	Specificity	Precision	F1 Score
Hole	94.9	98.74	96.02	95.48
Object	93.9	99.25	97.89	95.88
Oil spot	96.0	97.67	92.86	94.41
Thread error	96.7	98.19	94.59	95.63

Table 4
Overall model prediction results in analysis from the confusion matrix for dataset-1.

Accuracy	Error	Sensitivity	Specificity	Precision	F1_score
95.36	4.64	95.40	98.46	95.34	95.35

Table 5
Comparison of the deep networks.

Method	VGG16	Xception	Proposed ResNet
Weights parameters (in Million)	138	22	23.5
Validation Accuracy	88.76	93.10	95.36

Table 3 shows that the model is well-balanced and classifies each defect well. The specificity score is larger than the precision score, indicating that negative defects were classified well compared to false predictions. The overall model achieved 95.36% classification accuracy for all defects with a 4.64% error rate as shown in Table 4.

The proposed model results are validated by comparison with those of traditional CNN models, including the VGG16, ResNet50, and Xception networks in Table 5. The comparison shows that the proposed ResNet model performed better among all networks.

Finally, a comparison with state-of-the-art methods using deep networks is presented in Table 6.

A two-stage model using Inception 3 and a back propagation neural network (BPNN) was presented in [47]. In the first stage, an inception model was used for feature extraction, and in the second stage, the BPNN network was optimized using the Moath Flame optimization technique and used for classification. This method was tested on a dataset containing two types of defects, including holes and lines (horizontal and vertical) in fabric. However, the algorithm testing was limited to horizontal and vertical lines only.

Zhan et al. [48] approached the issue of imbalanced image distribution in a dataset by splitting the training set into support and query sets. The model was first trained using the support set and then fine-tuned using the query set. This technique resulted in the highest classification accuracy of 96.04% among the GoogleNet, ResNet, ResNetPreAct, and DenseNet networks. In [49], pretrained GoogleNet was proposed to classify woven fabric defects into six categories: holes, dark threads, spots, thread defects, switches off, and floods. The authors extracted features using GoogleNet, preprocessed images by histogram equalization and Gaussian filtering, and detected the fault location using the Fourier spectrum. Deep features from GoogleNet were then used to classify the flaws, achieving 94.46% accuracy.

In [50], a GoogleNet network was used to classify the defects from fabric surface images. They used a subset of Tilda Dataset i.e. woven fabric dataset in their experiment. The preprocessed images using Gaussian filter and Fourier transform. Later edge features are used to train GoogleNet for classification. In [51], the author created a dataset with images having regular texture regions. They categorized images into five defect types. They filtered noise from images using a Gaussian filter and pseudo-CNN. Later modified VGG network was used to classify defects. Their network has 62 Million learnable parameters and its classification rate was 93.92%. In [52], the five types of defects

Table 6
Comparison with state-of-art fabric defect classification models.

Model	Accuracy (%)
Few-shot learning [48]	96.04
Fourier Spectrum + GoogleNet [49]	94.46
GoogleNet [50]	93.60
Pseudo-CNN + Weight Initialized Adaptive Window [51]	93.92
Alexnet [52]	92.60
ResNet50 + KNN [53] (Binary classification)	95.88
Ensemble approach of CNN [54]	97.84
Proposed ResNet	95.36

including cuts, colors, holes, threads, and metal contamination from the MVTec Anomaly Detection dataset were used to train the Alexnet network and the model received 92.60% accuracy. In a study [53], a comparison of four CNN networks was presented to detect if the surface is defective or not. Among CNN, VGG16, ResNet50 and Inception network, ResNet50 performed well with 95.88% classification. Zhao et al. [54] used an ensemble approach to improve the classification rate of fabric defects. A subset of the Tilda dataset i.e. Germal Tilda dataset was used to validate the network and it received 97% and 96% precision and recall rate respectively and the cost of computation is large due to integration of the Densenet, Inception and Xception networks. The accuracy of the network and individual classification rate were not presented in the work.

The proposed model covers four different types of defects from dataset 1 that resemble the presence of objects and threads and ten defects from dataset 2. These defects are challenging to classify, but the model uses coarse features from the convolutional layer and fine-grained features from the residual layers to distinguish between them. As a result, the proposed model achieves higher or similar accuracy without requiring any preprocessing or additional feature extraction process. The weakness of dataset 2 is that each defect has unique texture background therefore model showed 100% accuracy.

5. Conclusions

This article presents an automated approach for classifying challenging fabric defects. The method involves using a deep neural network to classify defects from dataset-1 (roboflow-based fabric dataset containing four defect types) and dataset-2 (10 defects). One of the most ambiguous defects is oil spots, which can appear similar to other flaws on the fabric surface. Thus, the dataset is augmented with image processing techniques to provide more images. The augmentation allows a pre-trained ResNet network to be tuned effectively to differentiate between these challenging defect types. The residual layer of the network provides fine-grained details of texture patterns, allowing the network to identify subtle variations, irregularities, or inconsistencies within the texture pattern more precisely.

The proposed model has exhibited the following capabilities: (a) It can accurately classify with high specificity, resulting in low false recognition compared to state-of-the-art models. (b) Based on experimental results, the proposed network achieved an average accuracy of 95.36% and 95.35% F1-score for complex patterned cloths of dataset-1 and 100% accuracy for dataset-2. (c) The proposed method is more resistant to various sorts of printed materials. (d) The convolution neural network can distinguish between four types of defects. In the future, we plan to adopt a hybrid approach using a CNN to further improve the accuracy and validate the model with various datasets. Dataset 2 can be modified by generating synthetic images of various defects with similar textured backgrounds and further testing of the network is expected. Additionally, the model requires further optimization to meet the needs of the fabric industry.

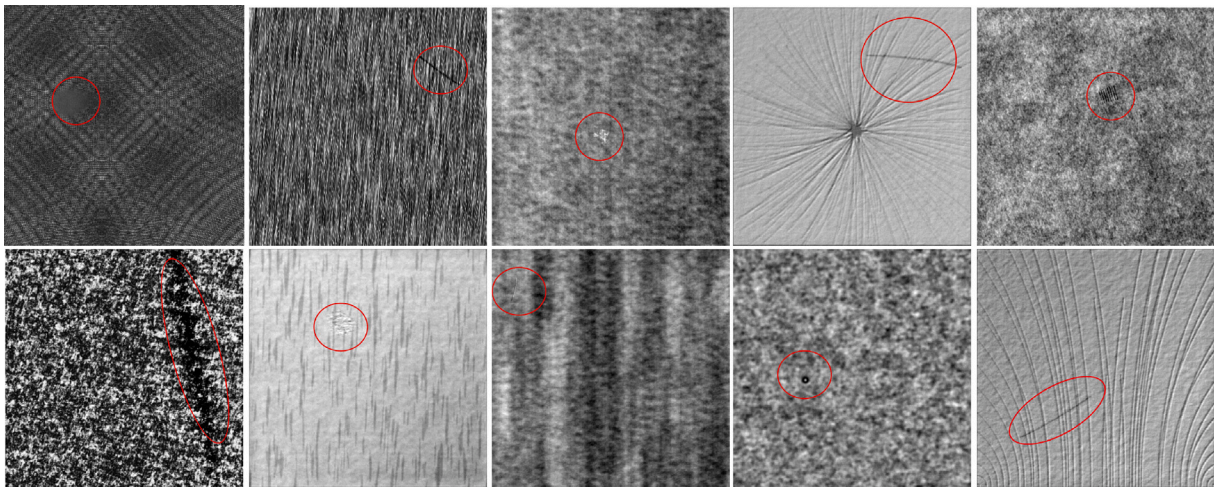

Fig. 16. Sample images of defect types from dataset-2.

CRediT authorship contribution statement

Hiren Mewada: Conceptualization, Methodology, Software. **Ivan Miguel Pires:** Data curation, Writing – original draft, Editing. **Pinalkumar Engineer:** Review, Formal analysis. **Amit V. Patel:** Review and modification for the final layout.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work is funded by FCT/MEC, Portugal through national funds and co-funded by FEDER—PT2020 partnership agreement under the project UIDB/50008/2020.

Appendix

Sample images of ten defects from dataset-2 are shown in Fig. 16. Dataset-2 provides labeling of defects but its location is not annotated. In Fig. 16, the location of the defect is annotated manually for demonstration only.

References

[1] M.C. Elemmi, B.S. Anami, N.N. Malvade, Defective and nondefective classif ication of fabric images using shallow and deep networks, Int. J. Intell. Syst. 37 (3) (2022) 2293–2318.
[2] A.S. Al-Waisy, D.A. Ibrahim, D.A. Zebari, S. Hammadi, H. Mohammed, M.A. Mohammed, R. Damaševičius, Identifying defective solar cells in electroluminescence images using deep feature representations, PeerJ Comput. Sci. 8 (2022) e992.
[3] Y. Huang, J. Jing, Z. Wang, Fabric defect segmentation method based on deep learning, IEEE Trans. Instrum. Meas. 70 (2021) 1–15.
[4] P.R. Jeyaraj, E.R.S. Nadar, Effective textile quality processing and an accurate inspection system using the advanced deep learning technique, Text. Res. J. 90 (9–10) (2020) 971–980.
[5] M.S. Biradar, B.G. Sheeparamatti, P.M. Patil, Fabric defect detection using competitive cat swarm optimizer based RideNN and deep neuro Fuzzy network, Sens. Imaging 23 (1) (2022) 3.
[6] M. Chen, L. Yu, C. Zhi, R. Sun, S. Zhu, Z. Gao, Z. Ke, M. Zhu, Y. Zhang, Improved faster R-CNN for fabric defect detection based on Gabor filter with Genetic Algorithm optimization, Comput. Ind. 134 (2022) 103551.
[7] T. Akiba, S. Sano, T. Yanase, T. Ohta, M. Koyama, Optuna: A next-generation hyperparameter optimization framework, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2019, pp. 2623–2631.
[8] Y. Tong, R. Chen, Improved cuckoo search algorithm for hyper-parameter optimization in deep learning and its application in activity recognition, 2023.
[9] J. Silvestre-Blanes, T. Albero-Albero, I. Miralles, R. Pérez-Llorens, J. Moreno, A public fabric database for defect detection methods and results, Autex Res. J. 19 (4) (2019) 363–374.
[10] RoboFlow, Fabric defect datasets, 2023, https://universe.roboflow.com. [Online Accessed 19 September 2023].
[11] DAGM 2007, DAGM 2007, 2007, https://paperswithcode.com/dataset/dagm2007. [Online Accessed 19 February 2024].
[12] R.M. Shashi, Fabric defect datasets, 2023, https://www.kaggle.com/datasets/rmshashi/fabric-defect-dataset. [Online Accessed 19 February 2024].
[13] J. Silvestre-Blanes, T. Albero-Albero, I. Miralles, R. Pérez-Llorens, J. Moreno, A public fabric image database for defect detection, 2019, https://content.sciendo.com/view/journals/aut/ahead-of-print/article-10.2478-aut-2019-0035.xml. [Online Accessed 19 February 2024].
[14] K. Vinayan, Aitex-fabric-image-database, 2020, https://www.kaggle.com/datasets/nexuswho/aitex-fabric-image-database. [Online Accessed 19 February 2024].
[15] H. Schulz-Mirbach, Aitex-fabric-image-database, 2020, https://lmb.informatik.uni-freiburg.de/resources/datasets/tilda.en.html. [Online Accessed 19 February 2024].
[16] Y. Qiu, Z. Zhou, J. Zhang, Evolving regularised random vector functional link by seagull optimisation algorithm for yarn-dyed fabric colour difference classification, in: Coloration Technology, Wiley Online Library.
[17] J. Li, W. Shi, D. Yang, Color difference classification of dyed fabrics via a kernel extreme learning machine based on an improved grasshopper optimization algorithm, Color Res. Appl. 46 (2) (2021) 388–401.
[18] S. Zhao, Y. Li, J. Zhang, J. Wang, R. Zhong, Real-time fabric defect detection based on multi-scale convolutional neural network, IET Collab. Intell. Manuf. 2 (2020) 189–196.
[19] G. Patil, S. Deshmukh, An approach to fabric defect detection using statistical methods for feature extraction, J. Phys. Conf. Ser. 2327 (1) (2022) 012033.
[20] H. Patel, H. Mewada, Dictionary properties for sparse representation: implementation and analysis, J. Artif. Intell. 11 (1) (2018) 1–8.
[21] X. Wang, B. Yan, R. Pan, J. Zhou, Real-time textile fabric flaw inspection system using grouped sparse dictionary, J. Real-Time Image Process. 20 (4) (2023) 1–15.
[22] H. Mewada, A.V. Patel, J. Chaudhari, K. Mahant, A. Vala, Composite fuzzy-wavelet-based active contour for medical image segmentation, Eng. Comput. 37 (9) (2020) 3525–3541.
[23] H.Y. Ngan, G.K. Pang, S.P. Yung, M.K. Ng, Wavelet based methods on patterned fabric defect detection, Pattern recognit. 38 (4) (2005) 559–576.
[24] H. Mewada, J. Al-Asad, A. Patel, J. Chaudhari, K. Mahant, A. Vala, Multi-channel local binary pattern guided convolutional neural network for breast cancer classification, Open Biomed. Eng. J. 15 (2021) 132–140.
[25] M. Jbene, A.D. El Maliani, M. El Hassouni, Fusion of convolutional neural network and statistical features for texture classification, in: 2019 International Conference on Wireless Networks and Mobile Communications, WINCOM, IEEE, 2019, pp. 1–4.
[26] M. Li, S. Cui, Z. Xie, et al., Application of Gaussian mixture model on defect detection of print fabric, J. Text. Res. 36 (8) (2015) 94–98.
[27] H. Mewada, R. Patel, S. Patnaik, A novel structure tensor modulated Chan–Vese model for texture image segmentation, Comput. J. 58 (9) (2015) 2044–2060.
[28] P. Bumrungkun, Defect detection in textile fabrics with snake active contour and support vector machines, J. Phys. Conf. Ser. 1195 (1) (2019) 012006.

[29] Z. Zhou, X. Yang, J. Ji, Y. Wang, Z. Zhu, Classifying fabric defects with evolving inception v3 by improved L2, 1-norm regularized extreme learning machine, Text. Res. J. 93 (3–4) (2023) 936–956.
[30] F.G. Yaşar Çiklaçandir, S. Utku, H. Özdemir, The effects of fusion-based feature extraction for fabric defect classification, Text. Res. J. (2023) 00405175231188535.
[31] T. Meeradevi, S. Sasikala, S. Gomathi, K. Prabakaran, An analytical survey of textile fabric defect and shade variation detection system using image processing, Multimedia Tools Appl. 82 (4) (2023) 6167–6196.
[32] L. Cheng, J. Yi, A. Chen, Y. Zhang, Fabric defect detection based on separate convolutional UNet, Multimedia Tools Appl. 82 (2) (2023) 3101–3122.
[33] C. Chen, X. Deng, Z. Yu, Z. Wu, Fabric defect detection using a one-class classification based on depthwise separable convolution autoencoder, J. Phys. Conf. Ser. 2562 (1) (2023) 012053.
[34] H. Zhao, T. Zhang, Fabric surface defect detection using SE-SSDNet, Symmetry 14 (11) (2022) 2373.
[35] Z. Zhou, W. Deng, Z. Zhu, Y. Wang, J. Du, X. Liu, Fabric defect detection based on feature fusion of a convolutional neural network and optimized extreme learning machine, Text. Res. J. 92 (7–8) (2022) 1161–1182.
[36] Y. Guo, X. Kang, J. Li, Y. Yang, Automatic fabric defect detection method using AC-YOLOv5, Electronics 12 (13) (2023) 2950.
[37] H. Yang, J. Jing, Z. Wang, Y. Huang, S. Song, YOLOV4-TinyS: a new convolutional neural architecture for real-time detection of fabric defects in edge devices, Text. Res. J. (2023) 00405175231202035.
[38] F. Li, F. Li, Bag of tricks for fabric defect detection based on Cascade R-CNN, Text. Res. J. 91 (5–6) (2021) 599–612.
[39] Z. Jia, Z. Shi, Z. Quan, M. Shunqi, Fabric defect detection based on transfer learning and improved faster R-CNN, J. Eng. Fibers Fabr. 17 (2022) 15589250221086647.
[40] H. Mewada, I.M. Pires, Electrocardiogram signal classification using lightweight DNN for mobile devices, Procedia Comput. Sci. 224 (2023) 558–564.
[41] A. Suryarasmi, C.C. Chang, R. Akhmalia, M. Marshallia, W.J. Wang, D. Liang, FN-Net: A lightweight CNN-based architecture for fabric defect detection with adaptive threshold-based class determination, Displays 73 (2022) 102241.
[42] Y. Huang, Z. Xiang, RPDNet: Automatic fabric defect detection based on a convolutional neural network and repeated pattern analysis, Sensors 22 (16) (2022) 6226.
[43] Z. Liu, C. Zhang, C. Li, S. Ding, Y. Dong, Y. Huang, Fabric defect recognition using optimized neural networks, J. Eng. Fibers Fabr. 14 (2019) 1558925019897396.
[44] S. Lin, Z. He, L. Sun, Self-transfer learning network for multicolor fabric defect detection, Neural Process. Lett. 55 (4) (2023) 4735–4756.
[45] F.G. Yaşar Çıklaçandır, S. Utku, H. Özdemir, Determination of various fabric defects using different machine learning techniques, J. Text. Inst. (2023) 1–11.
[46] S.S. Mohammed, H.G. Clarke, A hybrid machine learning approach to fabric defect detection and classification, in: International Congress of Electrical and Computer Engineering, Springer, 2022, pp. 135–147.
[47] N. Alruwais, E. Alabdulkreem, K. Mahmood, R. Marzouk, M. Assiri, A.A. Abdelmageed, S. Abdelbagi, S. Drar, Hybrid mutation moth flame optimization with deep learning-based smart fabric defect detection, Comput. Electr. Eng. 108 (2023) 108706.
[48] Z. Zhan, J. Zhou, B. Xu, Fabric defect classification using prototypical network of few-shot learning algorithm, Comput. Ind. 138 (2022) 103628.
[49] Y. Hu, G. Jiang, Weft-knitted fabric defect classification based on a Swin transformer deformable convolutional network, Text. Res. J. 93 (9–10) (2023) 2409–2420.
[50] R. Ashraf, Y. Ijaz, M. Asif, K.Z. Haider, T. Mahmood, M. Owais, et al., Classification of woven fabric faulty images using convolution neural network, Math. Probl. Eng. 2022 (2022).
[51] R. Sabeenian, E. Paul, C. Prakash, Fabric defect detection and classification using modified VGG network, J. Text. Inst. 114 (7) (2023) 1032–1040.
[52] N. Sandhya, N.M. Sashikumar, M. Priyanka, S.M. Wenisch, K. Kumarasamy, Automated fabric defect detection and classification: a deep learning approach, Text. Leath. Rev. 4 (2021) 315–335.
[53] R.A. Geze, A. Akbaş, Detection and classification of fabric defects using deep learning algorithms, Politeknik Dergisi 27 (1) (2023) 371–378.
[54] X. Zhao, M. Zhang, J. Zhang, Ensemble learning-based CNN for textile fabric defects classification, Int. J. Cloth. Sci. Technol. 33 (4) (2021) 664–678.