Optimized Gabor Filter Banks and Autoencoder Models for Enhanced Knitted Fabric Defect Detection

Shiva Bhandari[1], Shashidhar Ram Joshi[2], Sanjivan Satyal[3]

[1]PG Student, Department of Electronics and Computer Engineering, Pulchowk Campus, Tribhuvan University, Lalitpur, Nepal

[2]Dean of Engineering, Institute of Engineering, Tribhuvan University, Lalitpur, Nepal

[3]Assistant Professor, Department of Electronics and Computer Engineering, Pulchowk Campus, Tribhuvan University, Lalitpur, Nepal

Email: [1]078msice017.shiva@pcampus.edu.np, [2]srjoshi@ioe.edu.np, [3]sanziwans@pcampus.edu.np

Abstract

There is a high need for an automated system to detect fabric defects, as the current manual methods used in the garment industries in Nepal are unreliable and costly. Previous research has focused on specific fabric defects rather than overall fabric defects efficiently. This research employs two autoencoder models to identify different defects across different types of knitted fabrics, utilizing two datasets: the SFDG dataset and a custom dataset prepared from Butwal's garment industries. The models benefit from a carefully designed Gabor filter bank to examine fabric compositions. This filter bank is fine-tuned by modifying parameters, wavelength, and orientation to detect varieties of defects in knitted fabrics. These models get feature representations from the Gabor filter bank's outputs and help the system adapt to different types of defect patterns, making defect detection more reliable and accurate.

The nearest neighbor density estimator finds possible defects and marks on the fabric images. The model's effectiveness and strength are shown by validating it on different types of knitted fabrics, including plain and patterned fabrics, using evaluation metrices like cTPR and

DOI: https://doi.org/10.36548/jaicn.2024.3.001
Received: 20.05.2024, received in revised form: 14.06.2024, accepted: 02.07.2024, published: 17.07.2024
© 2024 Inventive Research Organization. This is an open access article under the Creative Commons Attribution-NonCommercial International (CC BY-NC 4.0) License

ROC AUC. The first model achieves a cTPR of 0.879 and an AUC score of 0.947, while the second model achieves a cTPR of 0.899 and an AUC score of 0.958.

Keywords: Autoencoder, Knitted Fabric Defect Detection, Gabor Filter, Nearest Neighbor Density Estimator

1. Introduction

The garment industry plays a significant role in Nepal's economic development. However, there is a notable gap in integrating Nepalese manufacturing industries with Industry 4.0 concepts, which needs urgent attention. In the garment industry, maintaining fabric quality is paramount as it directly impacts the brand's reputation. It is essential to mention that the scope of the research only includes knitted fabrics used for clothing in the garment industries.

1.1 Problem Definition

The garment industries in Nepal heavily rely on manual inspection for fabric defect detection, a process that is not only tedious and highly effortful but also prone to inconsistencies and human error. Existing automated systems often fall short due to their limited ability to detect a wide range of defects across different fabric textures. These systems are typically designed to identify specific types of defects, making them unsuitable for general application across various knitted fabrics. It is known that fabric defect detection during early stages of manufacturing is of much importance. This research focuses about early stage knitted fabric defect detection. The implementation of efficient and accurate fabric defect detection methods is essential for the Nepalese garment industry. This work proposes a solution to the problem of detecting defects in knitted fabrics using a combination of Gabor filter banks and machine learning techniques. Automatic fabric defect detection is inevitable but implementing such a system is a technical challenge due to two reasons. Firstly, fabric defects are of varying sizes and shapes. Additionally, there are significantly more error-free fabrics than error-prone fabrics. To address the variety of defects, multi-class detection systems have been developed. But these methods cannot work beyond certain classes of defects with good accuracy and have issues of unbalanced datasets. Secondly, background textures also vary in the patterns and they are of more than 70 kinds which depend on the weaving or knitting methods [13].

This research presents a novel approach to knitted fabric defect detection by combining optimized Gabor filter banks (by optimizing the orientation and bandwidth of the filter) with autoencoder architectures and nearest neighbor density estimator. Unlike previous works, such as (Zhou 2022), which focused on general fabric defects using a single Gabor filter and autoencoder mode [l5], our study introduces a dual autoencoder architecture tailored specifically for the diverse textures and defect patterns found in knitted fabrics. This dual approach enhances defect detection accuracy and reliability, particularly for complex knitted structures. Additionally, we provide a detailed analysis of parameter optimization for the Gabor filters, including a comprehensive exploration of wavelength and orientation adjustments, which has not been extensively covered in existing literature. Also, this research includes effectiveness of the approach by utilizing two different datasets.

1.2 Objectives

- To generate custom dataset as well as compare the performance of two different models for fabric defect detection.
- To develop a model that uses optimized Gabor filter bank, autoencoder and nearest neighbor density estimator for knitted fabrics inspection.
- To analyze the efficiency of the proposed approach in improving defect detection accuracy by comparing with other popular techniques used in fabric defect detection task.

1.3 Contribution of Research Work

This research work focuses on implementing model that uses Gabor filter bank as feature extraction and then applies autoencoder architecture network as feature learning step and finally uses nearest neighbour density estimator to detect defects in knitted fabrics only. Moreover analysis of two models only differing in autoencoder architecture network are done in standard dataset as well as self-prepared dataset from garment industries in Butwal city located in Terai region of Nepal. Performance of the model is done through two evaluation metrices cTPR and AUC (Area Under Curve). Similarly, effectiveness of the model is interpreted by comparing with other popular techniques implemented in fabric defect detection.

This research finds its scope in quality control during manufacturing process and helps to cover up huge economic loss.

2. Related Work

Over the years, efforts have been made to automate fabric defect detection. These works are extensively classified into four types: statistical, structural, model-based, and spectral approaches. The statistical approaches (D. a. Chetverikov 2002) (Latif-Amet 2000) hold first (mean or standard deviation) or second order statistics(correlation) for representing color information of the fabrics. However, relying solely on statistical data derived from unprocessed images, whether in color or grey-scale, proves insufficient in capturing the complete characteristics of fabric. This approach falls short in discerning between the distinct attributes of fabric and identifying defects, especially in instances involving patterns. The other approach structural one (D. Chetverikov 2000) is applicable for plain fabrics due to its simple patterns are easier to analyze and retrieve. Spectral methods analyze fabric textures based on frequency, which are less sensitive than methods that analyze textures based on spatial details. Machine vision-based techniques are also used in the fabric defect detection. Jing et al. [8] utilized a texture features extracted using Gabor filter and then utilized golden subtraction in order to segment the defects. In this technique, the patterned fabrics defects were not accurately segmented. Li et al.[12] proposed a low rank representation-based fabric detection method for the repeated texture structure of fabrics. However, the time complexity is a significant limitation in the method. It means computational complexity is also to be considered. Likewise, a fabric defect segmentation method by basic image and the Elo-algorithm is presented by Kang et al. [10] A novel method called IIER integrates the concept of the integral image into the Elo-rating algorithm to quickly detect defects in various types of fabric. Moreover, Guan et al. [5] proposed a method to detect fabric defects based on visual saliency, but it doesn't work for detecting defects in red-green-blue colored fabrics. Likewise, Ashraf et al. [1] proposed an image processing technique with CNN based GoogleNet that classifies defective and non-defective woven fabrics. Similarly, an automatic fabric defect detection method was proposed based on YOLOv5 algorithm (Jin 2021). A teacher-student architecture is defined to handle the shortage of fabric defect images. Zhou et al. [15] proposed a novel one class classification model for detecting variety of fabric defects based on applying optimized Gabor filter banks in SFDG dataset (Incorporated n.d.). This research is motivated by the works of Zhou et al. in order to detect defects in the knitted fabrics which uses optimized gabor filter bank and simple autoencoder architecture for fabric defect detection. The research is based on improving the performance of the method and application into optimizing the knitted fabric defect detection with custom dataset generated locally.

3. Proposed Work

The model works in two main stages: training and testing. It uses two types of fabric image datasets: the Standard Fabric Defect Glossary (SFDG) dataset and a custom dataset from garment industries in Butwal. During training, normal fabric images are inputted into the model. A Gabor filter bank with various orientations and bandwidths is applied to these images to create filtered images, which are then cropped into smaller patches. These patches undergo feature selection using an autoencoder architecture, generating D-dimensional feature vectors that are saved for later use in the testing stage.

In the testing stage, the model takes a raw image (either normal or defective) as input. It applies the same Gabor filter bank to this image, crops the filtered images into patches, and applies the same feature selection process as during training. The saved feature vectors from training are compared with the new feature vectors using a nearest neighbor density estimator to determine whether each patch is normal or defective. Finally, the identified defective patches are highlighted on the raw image. The Figure 1 shows the proposed system block diagram.

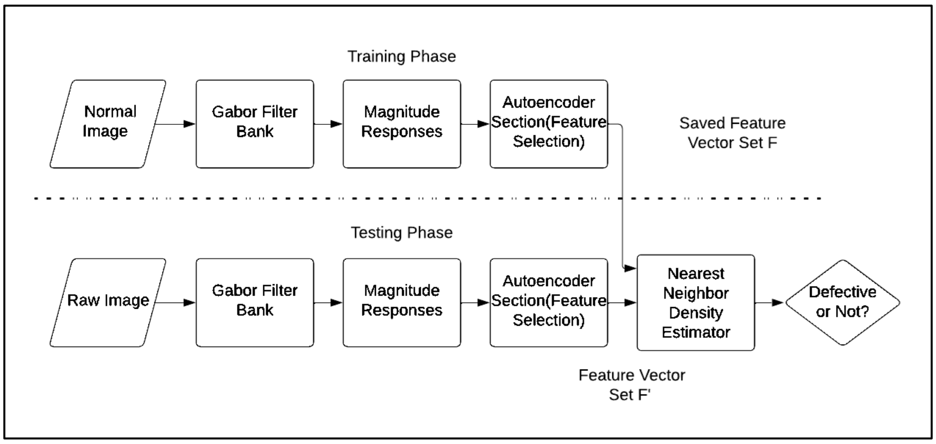

Figure 1. System Block Diagram

3.1 Related Theory

3.1.1 Gabor Filter Banks

A Gabor filter (Jones 1987) can be represented in the spatial domain as a product of a complex sinusoid and a Gaussian envelope. The sinusoidal component provides frequency selectivity, while the Gaussian envelope controls the spatial extent and orientation selectivity of the filter. The Gabor filter equation is given by:

$$G\ (x, y, \lambda, \theta, \psi, \sigma, y) = \exp((-x'^2 + \gamma^2 y'^2)/2\sigma^2)\cos((2\,\pi\, x')/\lambda\ +\ \psi) \ldots\ldots\ldots \quad (1)$$

- where $x' = x\cos\theta + y\sin\theta$ and $y' = -x\sin\theta + y\cos\theta$
- λ is the wavelength, θ is the angle of Gabor kernel direction, i.e., orientation parameter, ψ is the phase offset, γ is the spatial aspect ratio, σ is the standard deviation of the gaussian envelope,

In the fabric defect detection problem, the wavelength λ and orientation θ are crucial due to the varying orientations of fabrics and different yarn widths. The primary design challenge lies in identifying an optimal pair of λ and θ for knitted fabrics. This task is particularly demanding as it involves developing filter banks that perform effectively across various types of knitted fabrics. The wavelength λ in Gabor filters impacts the filter bank by influencing the frequency response and sensitivity to different spatial frequencies in the image. The orientation θ parameter in Gabor filters specifies the direction of the sinusoidal wave within the filter, influencing how the filter reacts to different features in an image.

3.1.2 Nearest Neighbor Density Estimator

This research applies a Nearest Neighbor Density Estimator (NNDE) designed to detect defective fabric patches during testing using feature vectors saved from the training phase. The nearest neighbor density estimator is a technique commonly used in fabric defect detection to estimate the density of defects or anomalies present in a fabric sample.

$$s_i = \frac{Dist(f_i',\ F')}{Dist(f_j,\ F)}, f_i' \in F', f_j \in F \ \ldots\ldots\ldots \quad (2)$$

where, f_i' represents feature vector of input image in testing phase whose feature vector set is F', f_j represents feature vector from training phase whose saved feature vector set is F, *Dist()* returns the distance between feature vector and its nearest neighbor in F.

Further the feature vector f_i' is to be classified into defective or non-defective based on the threshold τ as follows:

$$f_i' = \begin{cases} Defective, \ if \ s_i \geq \tau \\ Normal, \ otherwise \end{cases} \dots\dots\dots \tag{3}$$

- where, τ must be greater than or equal to one.

We divide the overall fabric defect detection process into following phases:

3.2 Collection of Knitted Fabric Data

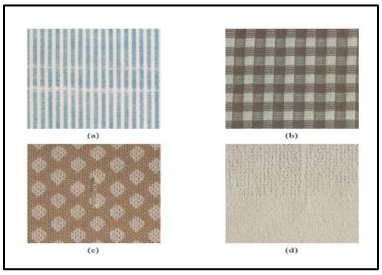

Figure 2. Various Fabric Images in SDFG Dataset

The collection of data for the model is done through two sources: one is from Standard Fabric Defect Glossary (SFDG) dataset and other is knitted fabric data collected from garment industries. The Figure 2 consists of images from SFDG dataset contains more complex fabric images compared to other simple datasets like the TILDA Textile Texture Database (University of Freiburg. Tilda Textile Texture Database. n.d.). It is a publicly available dataset. Each fabric image is a 512 x 512 size image. Differences between each image are seen in terms of fabric backgrounds, defects, colors, etc. Both plain and patterned fabric images are included. It consists of 1940 image samples in TIF format including rotated as well as non-rotated images.

In case of custom dataset (Figure 3), fabric image collection is done through visiting various garment industries in Butwal, Nepal. The images are captured. There are 1156 images in custom dataset. It is important to notice that most of the collected images do not contain defects.

Figure 3. Various Fabric Images from Industrial Custom Dataset

The captured images are taken during knitting process but not after they are being manufactured. It is obvious that data collected from this method needs more preprocessing. In order to simulate defects present in real world scenarios and help in training to classify the image patches as normal or defective, synthetic dataset is generated using image processing techniques to create synthetic defects such as blobs (circular defects with random radius), scratches (line defects of various thickness) and noise on the fabric image as in Figure 4.

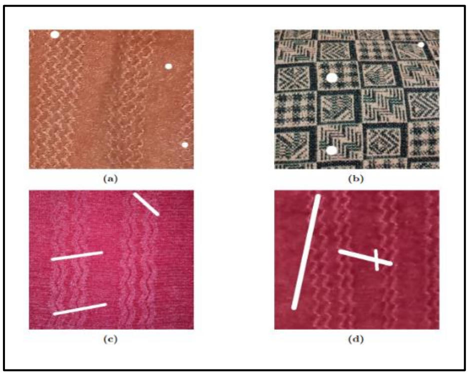

Figure 4. Various Fabric Images Generated via Synthetic Method

3.3 Data Preprocessing

Data preprocessing is an essential step to prepare raw fabric images for defect detection. Several preprocessing techniques, as shown in Figure 5 were employed to prepare the fabric images for defect detection:

- **Resizing:** The industrial custom data that was collected is first converted to fixed size of 512 x 512 pixels. This is done as the standard dataset is used in this very size. Images are then resized to the patch size specified (i.e., 32x32 pixels).

- **Normalization:** Pixel values are normalized to the range [0, 1] to standardize the input data.

- **Patch Extraction:** Large images are divided into smaller patches to focus on localized defects. Patches are extracted with a specific size (32) and stride (10) to ensure all areas are covered.

- **Gabor Filter Application:** Gabor filters are created with specific parameters (lambda, theta, gamma, sigma, psi). These filters are applied to the images to extract texture features. The detailed explanation is explained in subsection 3.1.1.

- **Feature Vector Generation:** Feature vectors are generated from the filtered images using mean and standard deviation values.

- **Randomization and Splitting:** The patches are shuffled to ensure randomization. Then the dataset is split into training set (80%) and testing set (20%).

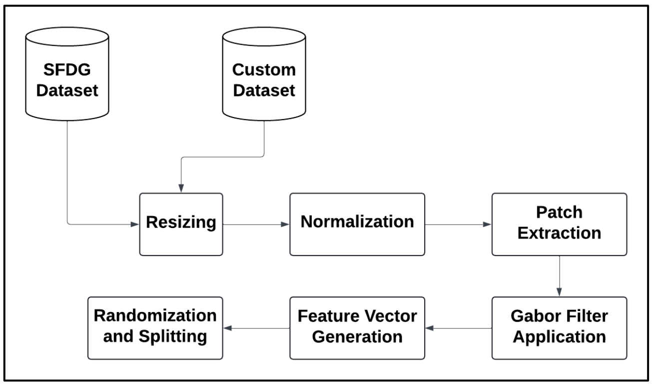

Figure 5. Data Preprocessing steps of Datasets used

3.4 Gabor Filter Bank Optimization

In the problem of detecting defects in knitted fabrics, the wavelength (λ) and orientation (θ) are crucial factors. The challenge lies in identifying an optimal pair of λ and θ suitable for all types of knitted fabrics. During the feature selection phase, the magnitude response of the Gabor filter is especially important.

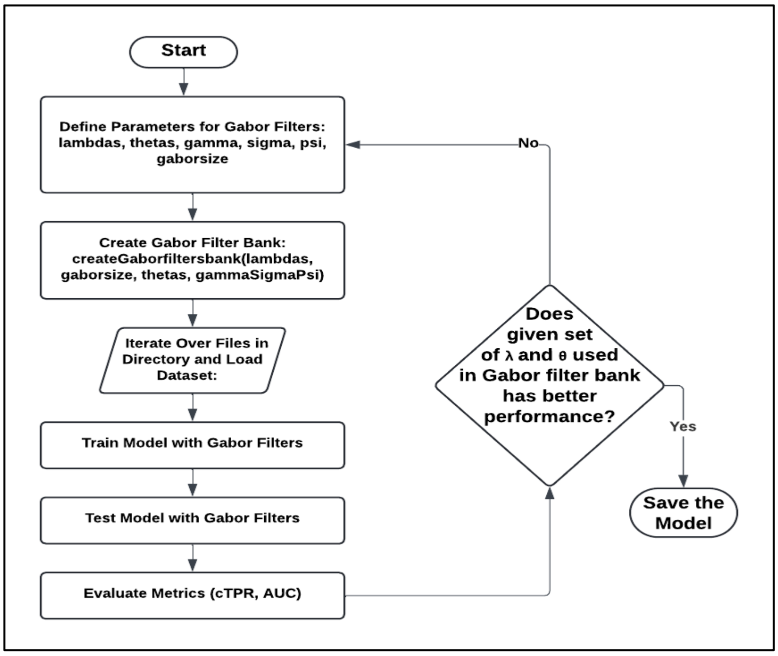

Figure 6. Flow of Gabor Filter Bank Optimization

A gabor filter bank is selected based on different wavelengths and bandwidths that can almost cover all types of knitted fabrics. To apply the Gabor filter bank, we implemented various functions. The optimization flow of the Gabor filter bank is depicted in Figure 6. To determine the optimal orientation parameter, a range of angles is considered to be from 0 and 180 degrees and intervals are set as mentioned in Figure 15. Similarly, various wavelengths are chosen and evaluated for performance as depicted in Figure 14. The number of filters in the filter bank is the product of the number of wavelengths and orientations used. This method creates a Gabor filter bank applied to detect defects in knitted fabrics. Experiments with different wavelengths and orientations were conducted and their performances was evaluated.

3.5 Feature Selection: Use of Autoencoder Architecture

Autoencoders are not directly applicable to fabric images due to the lack of large-scale datasets. Therefore, two different types of autoencoder models were implemented and compared. The selected features included texture patterns captured by the Gabor filters. Each feature vector consisted of the magnitude responses from the filtered images. The model was implemented using Python and the PyTorch library. Performance was evaluated using metrics such as Conditioned True Positive Rate (cTPR) and Area Under the Curve (AUC) from Receiver Operating Characteristic (ROC) analysis. Each model consists of two different architecture based on the boolean value of gabor parameter. The two architecture serve different purposes:

3.5.1 Gabor Features + Fully Connected Layers

This architecture is designed to work with feature vectors extracted using Gabor filters. It captures more complex relationships between features by using multiple fully connected layers. This is suitable for datasets where Gabor features effectively highlight defects.

3.5.2 Convolutional Layers

This architecture uses 1D convolutional layers to process raw data directly. Convolutions are particularly effective in capturing spatial hierarchies and local patterns, making this model suitable for datasets where spatial relationships within the data are critical for detecting defects.

3.5.3 Model 1

The autoencoder class is initialized with two key parameters: dim_in, representing the dimension of the input data, and gabor, a boolean indicating whether Gabor filters will be used in the encoder. Within the constructor, dim_in is stored as both the input and output dimension, and the gabor flag is retained for future reference. The Figure 7 depicts the model summary of autoencoder when Gabor parameter is True.

```
----------------------------------------------------------------
Layer (type)               Output Shape         Param #
================================================================
Linear-1                   [-1, 90]             11,610
BatchNorm1d-2              [-1, 90]             180
Sigmoid-3                  [-1, 90]             0
Linear-4                   [-1, 64]             5,824
Linear-5                   [-1, 32]             2,080
ReLU-6                     [-1, 32]             0
Linear-7                   [-1, 64]             2,112
Linear-8                   [-1, 90]             5,850
Linear-9                   [-1, 128]            11,648
BatchNorm1d-10             [-1, 128]            256
Sigmoid-11                 [-1, 128]            0
================================================================
Total params: 39,560
Trainable params: 39,560
Non-trainable params: 0
----------------------------------------------------------------
```

Figure 7. The Model Summary of Autoencoder Architecture when Gabor Parameter is True

If Gabor is set to False, the encoder uses 1D convolutional layers followed by ReLU activations and max-pooling and decoder uses transposed 1D convolutional layers followed by ReLU activations. The Figure 8 depicts the model summary of autoencoder when Gabor parameter is False.

Layer (type)	Output Shape	Param #
Conv1d-1	[-1, 16, N/45]	1,456
ReLU-2	[-1, 16, N/45]	0
Conv1d-3	[-1, 8, (N-4)/3]	648
ReLU-4	[-1, 8, (N-4)/3]	0
MaxPool1d-5	[-1, 8, (N-6)/3]	0
ConvTranspose1d-6	[-1, 16, (N-6)/3*3]	656
ReLU-7	[-1, 16, (N-6)/3*3]	0
ConvTranspose1d-8	[-1, 2, N]	1,442
ReLU-9	[-1, 2, N]	0

Total params: 4,202
Trainable params: 4,202
Non-trainable params: 0

Figure 8. The Model Summary of Autoencoder Architecture when Gabor Parameter is False

3.5.4 Model 2

The Figure 9 depicts the model summary of autoencoder when Gabor parameter is True.

Layer (type)	Output Shape	Param #
Linear-1	[-1, 256]	32,896
BatchNorm1d-2	[-1, 256]	512
ReLU-3	[-1, 256]	0
Dropout-4	[-1, 256]	0
Linear-5	[-1, 128]	32,896
BatchNorm1d-6	[-1, 128]	256
ReLU-7	[-1, 128]	0
Dropout-8	[-1, 128]	0
Linear-9	[-1, 90]	11,610
BatchNorm1d-10	[-1, 90]	180
ReLU-11	[-1, 90]	0
Linear-12	[-1, 128]	11,648
BatchNorm1d-13	[-1, 128]	256
ReLU-14	[-1, 128]	0
Linear-15	[-1, 256]	33,024
BatchNorm1d-16	[-1, 256]	512
ReLU-17	[-1, 256]	0
Linear-18	[-1, 128]	32,896

Total params: 156,686
Trainable params: 156,686
Non-trainable params: 0

Figure 9. The Model Summary of Autoencoder Architecture when Gabor Parameter is True

The forward method in both Model 1 and Model 2 defines the forward pass through the autoencoder. It takes an input x. The input is passed through the encoder to obtain the code.

The code is then passed through the decoder to reconstruct the input. The reconstructed input and the code are returned as output.

If Gabor is set to False, the encoder uses 1D convolutional layers followed by ReLU activations and max-pooling and the decoder consists of transposed 1D convolutional layers followed by ReLU activations. The Figure 10 depicts the model summary of autoencoder when Gabor parameter is False.

```
----------------------------------------------------------------
Layer (type)               Output Shape         Param #
================================================================
Conv1d-1                   [-1, 64, 64]         448
ReLU-2                     [-1, 64, 64]         0
BatchNorm1d-3              [-1, 64, 64]         128
Conv1d-4                   [-1, 128, 32]        24,704
ReLU-5                     [-1, 128, 32]        0
BatchNorm1d-6              [-1, 128, 32]        256
MaxPool1d-7                [-1, 128, 16]        0
Conv1d-8                   [-1, 256, 8]         98,560
ReLU-9                     [-1, 256, 8]         0
BatchNorm1d-10             [-1, 256, 8]         512
MaxPool1d-11               [-1, 256, 4]         0
ConvTranspose1d-12         [-1, 128, 8]         98,432
ReLU-13                    [-1, 128, 8]         0
BatchNorm1d-14             [-1, 128, 8]         256
ConvTranspose1d-15         [-1, 64, 16]         24,576
ReLU-16                    [-1, 64, 16]         0
BatchNorm1d-17             [-1, 64, 16]         128
Conv1d-18                  [-1, 2, 16]          386
ReLU-19                    [-1, 2, 16]          0
================================================================
Total params: 248,386
Trainable params: 248,386
Non-trainable params: 0
----------------------------------------------------------------
```

Figure 10. The Model Summary of Autoencoder Architecture when Gabor Parameter is False

3.6 Training Phase

Here, the model takes normal images as input and applies the aforementioned Gabor filter bank. The filtered images are then cropped into small patches, and feature selection based on the previously described autoencoder architecture is applied to these patches. Feature vectors are generated and saved for the testing phase. The autoencoder's goal during training is to minimize the reconstruction error between the input data and the reconstructed data, achieved by optimizing mean squared error (MSE) loss. The Adam optimizer updates the model parameters based on the computed loss. The training process involves iterating over the training dataset, performing forward and backward passes, and updating the model parameters until convergence is reached.

3.7 Testing Phase

Here, a raw image (either defective or normal image) is taken by the model and same Gabor filter bank is applied and filtered images are cropped and then same autoencoder architecture is applied to the image for feature selection. Finally, the nearest neighbor density estimator method is utilized which takes the saved feature vector and new feature vector to determine an image patch is either normal or defective.

4. Results and Discussion

To ensure the effectiveness of the two models proposed in this paper, various experiments are conducted on widely used public dataset of SDFG as well as custom dataset. The datasets are knitted fabric images of size 512 x 512 pixels. The complete experiment were carried on a local PC. The operating system (OS) was Windows 10 Enterprise Edition, and the hardware was Intel(R) Core(TM) i5-5200U CPU @ 2.20GHz, NVIDIA GeForce and 8 GB RAM.

4.1 Model 1 Results and Analysis

The Figure 11 output shows the input defective images and how the knitted fabrics defects are overlayed on the input image for plain as well as patterned knitted fabrics using model 1. Here the dataset used is SDFG dataset. This output is from the model having best attained evaluation metrices. The Figure 12 output shows the input defective images and how the knitted fabrics defects are overlayed on the input image for plain as well as patterned knitted fabrics using model 1 on the custom dataset. Similarly for the case of custom made knitted fabrics images, the defect detection performance is similar in effectiveness as in using SDFG dataset. This can also be validated using evaluation metrices comparison for plain and patterned cases.

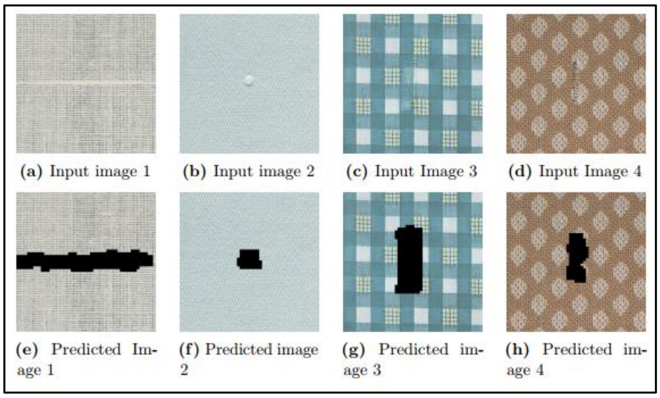

Figure 11. Output of Plain and Patterned Image Knitted Fabric Defect Detection using Model 1 on SFDG Dataset

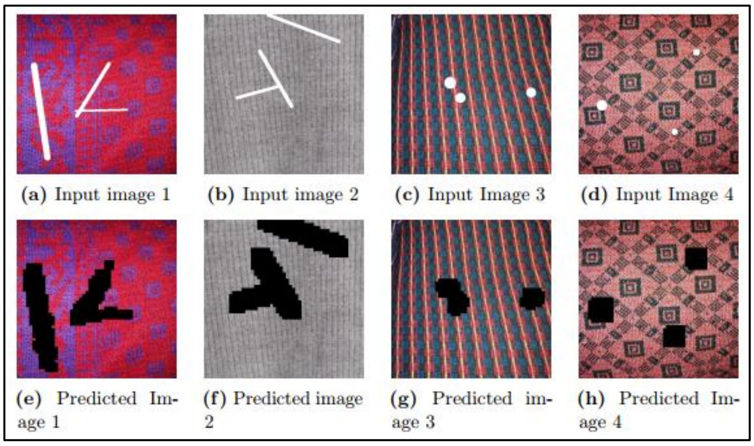

Figure 12. Output of Plain and Patterned Image Knitted Fabric Defect Detection using Model 1 on Custom Dataset

As from Table 1, it can be deduced that the model performs better for SFDG dataset than custom dataset. It is also observable that model performance for plain knitted fabrics while it slightly degrades for patterned knitted fabrics images. The cTPR and AUC values are comparatively higher for plain knitted fabrics while the values of cTPR and AUC are slightly less for patterned knitted fabrics. The Figure 13 demonstrates visualization of evaluation metrices instance in the model 1.

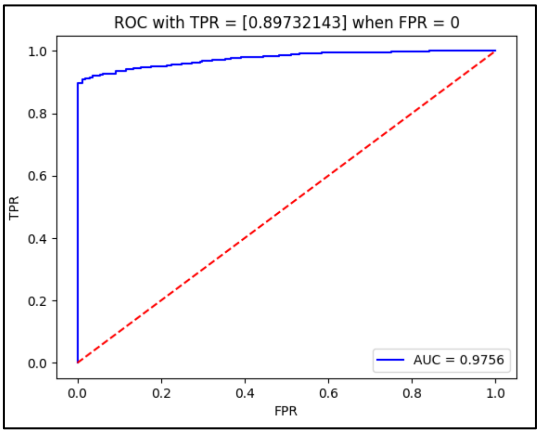

Figure 13. Evaluation of Model Indicating cTPR and AUC for Particular Image in Model 1

Table 1. Performance Impact of Number of Epochs and Patch Size on Evaluation Metrices taken from SDFG Dataset as well as Custom Dataset for Model 1

	Epochs = 1000, Patch size = 32	**Epochs = 1000, Patch size = 16**	**Epochs = 1000, Patch size = 64**
cTPR knitted (SFDG)	0.879	0.778	0.815
AUC knitted (SFDG)	0.941	0.872	0.901
cTPR knitted (Custom)	0.871	0.768	0.806
AUC knitted (Custom)	0.943	0.866	0.898

From the above Figure 13 and Table 1 it is clear that both parameters, i.e., cTPR and ROC AUC both have significant impact and when both scores are high then defects are better detected in image. Similarly, the effect of parameters like patch size and number of epochs have significant effects in the performance of the model. It can be illustrated from the Table 1.

4.2 Model 2 Results and Analysis

The visual outputs from this model are similar in both types of datasets used with that of model 1. It is to be known that model 2 has comparatively better performance than model 1 which will be illustrated in upcoming sections. Quantitatively, this can be validated using evaluation metrices comparison for plain and patterned cases. Additionally, from Table 2, it can be deduced that the model 2 has similar trends as in model 1 for SFDG and custom datasets. The cTPR and AUC values are comparatively higher for plain knitted fabrics while the values of cTPR and AUC are slightly less for patterned knitted fabrics. Similarly the effect of parameters like patch size and number of epochs have significant effects in the performance of the model. It can be illustrated from the Table 2.

Table 2. Performance Impact of Number of Epochs and Patch Size on Evaluation Metrices taken from SDFG Dataset as well as Custom Dataset for Model 2

	Epochs = 1000, Patch size = 32	**Epochs = 1000, Patch size = 16**	**Epochs = 1000, Patch size = 64**
cTPR all knitted (SDFG)	0.899	0.789	0.869
AUC all knitted (SDFG)	0.958	0.878	0.910
cTPR all knitted (Custom)	0.875	0.778	0.801
AUC all knitted (Custom)	0.953	0.873	0.890

4.3 Comparison between the Two Models Implemented

In order to compare the two models used, we summarize the cTPR and AUC metrices of the models after they are fined tuned and tested. The following Table 3 summarizes the comparison between the two models. It is clear from the table that cTPR and AUC obtained from model 2 is higher than those obtained from model 1. This suggests that comparatively model 2 has better fabric defect detection capabilities than model 1.

4.4 Comparison of Models with Other Feature Learning Techniques

The effectiveness of the models is compared with two other popular feature learning techniques used in fabric defect detection system. One of them is the Handcrafted model (D. a. Chetverikov 2002) that takes features from mean and standard deviation of the provided data. Next is the Principal Component Analysis (PCA) method (Beirão 2004). The evaluation metrices are generated and recorded for comparison between the applied model in this thesis work and these two techniques. The Table 3 provides insight of comparison between the two different models used with other feature learning techniques.

Table 3. Performance Comparison among Various Feature Learning Techniques

	Model 1	**Model 2**	**PCA**	**Handcrafted Model**
cTPR (SDFG)	0.879	0.899	0.789	0.790
AUC (SDFG)	0.947	0.958	0.921	0.926
cTPR (Custom)	0.870	0.875	0.776	0.782
AUC (Custom)	0.943	0.953	0.918	0.921

4.5 Impact of Gabor Filter Parameters

As number of wavelengths and orientations are increased the number of gabor filters in filter bank increases. It will be clear in following subsections that increasing the number of wavelengths and orientations increase the cTPR metrics but it also increase the processing time of the system.

4.5.1 Impact of Wavelength Parameter

It is important to note that changing a single value of the wavelength parameter alters the total number of filters by the number of orientations used. For example, if 5 wavelengths and 20 orientations are used, there will be 100 Gabor filters. Reducing the number of wavelengths to 4 decreases the total filters by 20. Increasing the number of wavelengths improves the cTPR of the model but also raises the processing time, creating a trade-off between cTPR and processing speed. Figure 14 illustrates the impact of wavelengths on the system's cTPR. Increasing more number of wavelengths, there is also increase in cTPR of the model but it also has impact in time complexity as processing time increases. Thus there is trade-off between cTPR and processing speed. After performing a number of experiments where the wavelength parameter was varied while keeping the orientation parameter constant, a total of 5 wavelengths were selected for use.

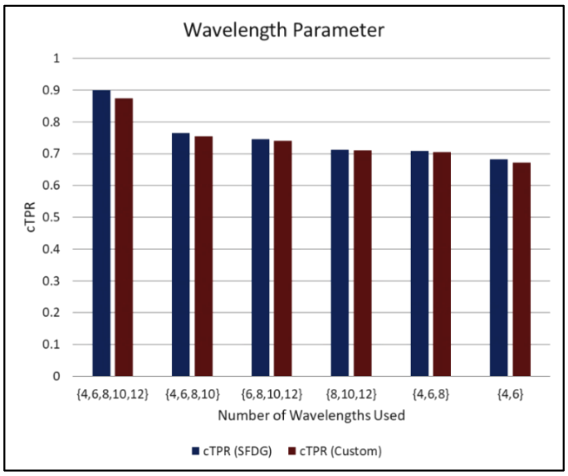

Figure 14. Performance Impact of Gabor Filter Bank Parameter: Wavelength on cTPR

4.5.2 Impact of Orientation Parameter

It is to be noted that changing single value of orientation parameter changes the total number of filters used by the number of wavelengths used. For instance if total number of orienations used is 20 and number of wavelengths used is 5 then total number of gabor filters used will be 100. When we change this number of orientations to 19 then total filters used will be reduced by 5. Increasing more number of orientations, there is also increase in cTPR of the model but it also has impact in time complexity as processing time increases. Thus there is trade-off between cTPR and processing speed. The following Figure 15 illustrates the impact of orientations in cTPR of the system. After number of experiments performed changing orientation parameter but keeping wavelength parameter constant, total number of orientation used is selected to be 20.

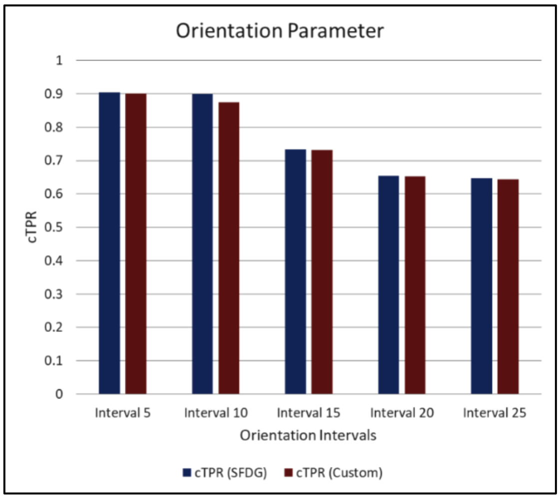

Figure 15. Performance Impact of Gabor Filter Bank Parameter: Orientation on cTPR

5. Conclusion

The research demonstrates that implemented models are effective in knitted fabric defect detection. In this research work, we proposed two different models each one having separate autoencoder architecture for feature learning and similar yet specially designed Gabor filter bank and nearest neighbor density estimator that detects knitted fabric defects. One of the advantage with our system is that it does not require defective samples for detection purpose. In order to illustrate the effectiveness of the system, we used two separate datasets for the research, one being SDFG dataset and another being custom dataset from images taken from local garment industries. The two different autoencoder architectures are implemented for comparison purpose. The overall performance of the system is measured by cTPR and AUC scores. The results obtained from Model 2 are better in knitted fabric defect detection as indicated by evaluation metrices used. Similarly, we compared our work or models implemented with other two popular methods for feature learning i.e., Handcrafted model and PCA method. The comparison shows that even though Model 1 and Model 2 have slightly higher AUC scores than other two methods, there is significant improvement of cTPR scores for model 1 and model 2 than other techniques. To further illustrate the applicability of the models implemented, they are analyzed for various types of knitted fabrics including plain, patterne4.

References

[1] Ashraf, Rehan, Yasir Ijaz, Muhammad Asif, Khurram Zeeshan Haider, Toqeer Mahmood, and Muhammad Owais. "Classification of Woven Fabric Faulty Images Using Convolution Neural Network." Mathematical Problems in Engineering 2022, no. 1 (2022): 2573805.

[2] Beirão, Céu L., and Mário AT Figueiredo. "Defect detection in textile images using Gabor filters." International Conference on Image Analysis and Recognition (Springer Berlin Heidelberg), 2004: 841-848.

[3] Chetverikov, Dmitry. "Pattern Regularity as a Visual Key." Image and Vision Computing 18, no. 12 (2000): 975–985.

[4] Chetverikov, Dmitry, and Allan Hanbury. "Finding defects in texture using regularity and local orientation." Pattern Recognition 35, no. 10 (2002): 2165-2180.

[5] Guan, Shengqi, and Hongyu Shi. "Fabric defect detection based on the saliency map construction of target-driven feature." The Journal of the Textile Institute 109, no. 9 (2018): 1133-1142.

[6] Incorporated, Cotton. Cotton Incorporated. Standard Fabric Defect Glossary. n.d. https://www.cottoninc.com/quality-products/textile-resources/fabric-defect-glossary/ (accessed May 2023).

[7] Jin, Rui, and Qiang Niu. "Automatic fabric defect detection based on an improved YOLOv5." Mathematical Problems in Engineering 2021, no. 1 (2021): 7321394.

[8] Jing, Junfeng, Panpan Yang, Pengfei Li, and Xuejuan Kang. "Supervised defect detection on textile fabrics via optimal Gabor filter." Journal of Industrial Textiles 44, no. 1 (2014): 40-57.

[9] Jones, Judson P., and Larry A. Palmer. "An evaluation of the two-dimensional Gabor filter model of simple receptive fields in cat striate cortex." Journal of Neurophysiology 58, no. 6 (1987): 1233-1258.

[10] Kang, Xuejuan, and Erhu Zhang. "A universal defect detection approach for various types of fabrics based on the Elo-rating algorithm of the integral image." Textile Research Journal 89, no. 21-22 (2019): 4766–4793.

[11] Latif-Amet, A., Aysin Ertüzün, and Aytül Erçil. "An efficient method for texture defect detection: sub-band domain co-occurrence matrices." Image and Vision Computing 18, no. 6-7 (2000): 543–553.

[12] Li, Peng, Junli Liang, Xubang Shen, Minghua Zhao, and Liansheng Sui. "Textile fabric defect detection based on low-rank representation." Multimedia Tools and Applications 78 (2019): 99-124.

[13] Ngan, Henry YT, Grantham KH Pang, and Nelson HC Yung. "Automated fabric defect detection—A review." Image and vision computing 29, no. 7 (2011): 442-458.

[14] University of Freiburg. Tilda Textile Texture Database. n.d. https://lmb.informatik.uni-freiburg.de/resources/datasets/tilda.en.html (accessed May 2023).

[15] Zhou, Hao, Yixin Chen, David Troendle, and Byunghyun Jang. "One-class model for fabric defect detection." arXiv preprint arXiv:2204.09648 (2022).