CHAPTER 5

AUGMENTING FINGER VEIN IMAGE DATASETS

5.1 INTRODUCTION

In machine learning and artificial intelligence, dataset augmentation is considered important, especially when trying to enhance the models' performance and robustness. It creates newer training samples out of data by performing distinct transformations, thereby adding value to the scope. This reduces the risks of overfitting in a model, where the model fits well with training data but struggles with unseen data after a diverse range of training is simulated. AI and ML adopt dataset augmentation as an essential method to enhance model performance as well as model robustness. The method creates new training examples from available data through different transformation techniques, which expand both the data volume and its diversity. The technique reduces overfitting by making models encounter different variations through this process (Nagaraju M. et. al. 2022).

The FV image databases contain images of different persons but provide only a few images for each individual. Training using insufficient samples makes it harder for the model to learn vein patterns properly, thus reducing its verification accuracy and generalization capabilities. The training process benefits from a diverse image collection that includes variations in rotation positions, displacement, and lighting conditions. The model learns consistent patterns across different conditions because of the variability in the data. Training using a larger dataset allows the model to achieve better generalization and accuracy.

5.2 STRATEGIES FOR AUGMENTATION

The research employs two main augmentation approaches, which include conventional transformations and Generative Adversarial Networks (GANs). Transformations using conventional approaches depend on pre-established rules and geometric transformations to introduce variety into design systems in a simple manner and with moderate resource requirements. GANs establish a DL-based approach to create realistic images alongside diverse outputs. The combination of images produced by both methods creates a robust dataset that optimizes training efficiency for DL models (Shorten C. & Khoshgoftaar T. M., 2019.)

5.2.1 Augmentation using Conventional Transformation

Conventional data augmentation methods use established geometric rules to perform transformations through rotation, translation, scaling, and flipping. The methods provide easy access to diverse applications since they require fewer resources and simpler implementation. Dataset variability through conventional augmentation methods depends on the defined transformation, even though the extent of variability is limited by the nature of the transformations. The transformation operations applied in this work are rotation, translation, scaling, flipping, brightness adjustment and contrast adjustment. These basic transformation techniques enhance model generalization by exposing the model to different versions of the original data.

Rotation produces images at various viewing angles to address the minor variations in finger positioning during capture. As a result, the model learns to improve its ability to correctly identify images regardless of the angle at which the finger is positioned.

This transformation is useful when consistent finger alignment cannot be guaranteed. Rotation is mathematically denoted using the equation 5.1:

$$x' = x * \cos(\theta) - y * \sin(\theta) \quad \text{and} \quad y' = x * \sin(\theta) + y * \cos(\theta) \qquad (5.1)$$

Here, ‘x’ and ‘x'’ refer to the horizontal pixel positions before and after rotation, respectively. Similarly, ‘y’ and ‘y'’ are the original and rotated vertical pixel coordinates. θ is the angle of rotation, which can vary between -15 and 15 degrees.

Shifting corresponds to moving the image horizontally or vertically. Shifting operation helps the system handle small changes in the finger’s position while capturing vein images. Adding shifted images to the training dataset helps the model to recognize vein patterns even if the finger is not perfectly centred. This improves the system’s ability to deal with minor positional differences. The mathematical representation of shifting is shown in the equation 5.2:

$$x' = x + dx \quad \text{and} \quad y' = y + dy \qquad (5.2)$$

Here, x, x' refer to the original and shifted horizontal pixel positions, respectively. Similarly, y and y' are the original and shifted vertical pixel positions, respectively. The amounts by which the image is shifted horizontally and vertically are

denoted by dx and dy, respectively. These shift values can vary within a range of -10 to 10 pixels.

Various lighting conditions are simulated by applying variations in brightness. This allows the model to adjust to changes in ambient light. The model thus learns to recognize fluctuations in lighting, making the system more reliable under different illumination conditions. Brightness variation applied is represented using the equation 5.3:

$$np = op + bf \tag{5.3}$$

where np and op are the new and old pixel values and bf is the brightness factor, which varies within the range -0.2 and 0.2.

5.2.2 Augmentation using Generative Adversarial Networks

Generative Adversarial Network (GANs) is a DL algorithm that can be used for dataset augmentation (Zhang J., et al., 2019). GANs can produce synthetic FV images that resemble actual images, thereby improving the diversity of the dataset. GANs require additional computational resources and have complicated implementations. However, GANs can create images that are more realistic and highly diverse when compared to simple geometric transformations. It can create a range of new images by capturing intricate features and variations that are present in the original finger vein dataset. This is achieved by training a generator for producing new data samples that mimic real data, along with a discriminator responsible for identifying real and generated samples.

5.2.2.1 Architecture of GAN

GAN consists of a discriminator and a generator. The generator creates fake images while the discriminator judges the created images. These models undergo adversarial training, which allows the generator to compete and thereafter create heavily realistic images that would trick the discriminator (Goodfellow I. et. Al. 2014). A typical GAN architecture is illustrated in Figure 5.1.

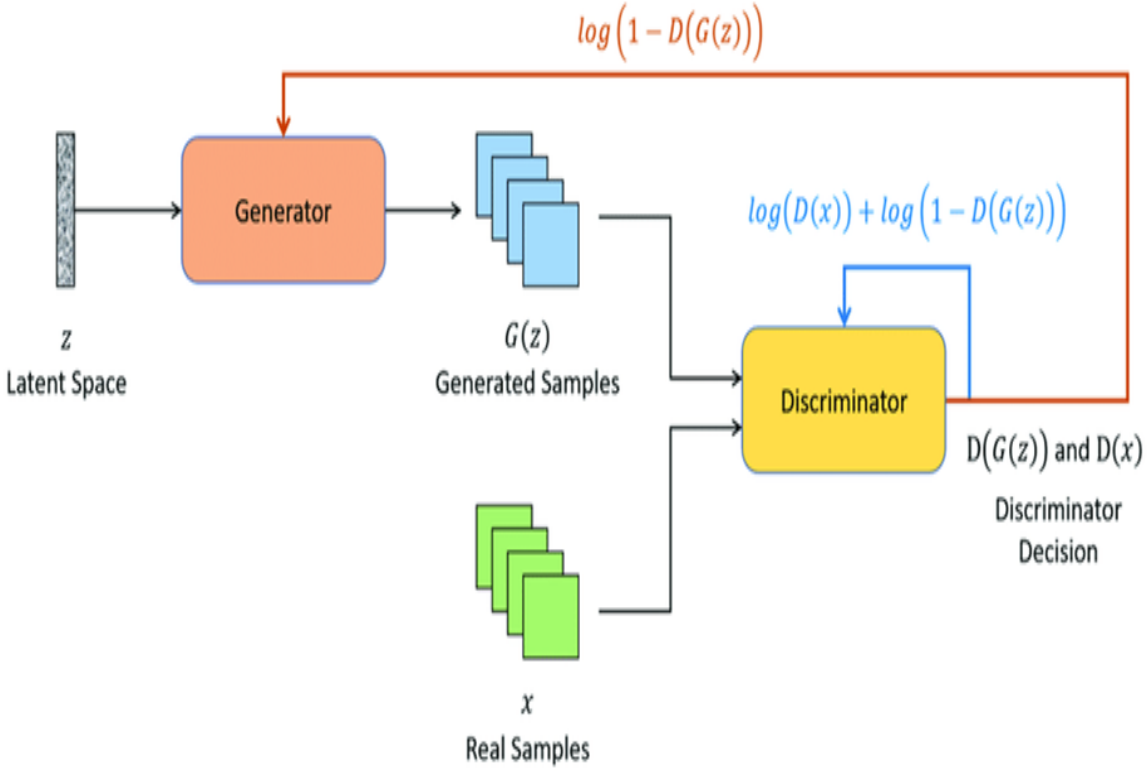


Figure 5.1 Typical GAN Architecture (Goodfellow I. et. al 2014)

5.2.2.2 Architecture of Conditional GAN

This work uses a Conditional Generative Adversarial Network (cGAN) for FV image augmentation. In cGANs the image generation process can be conditioned on specific classes using the labelled vein images. This ensures that the augmented images contain the original specific vein patterns, preserving their semantic meaning. Thus, cGANs can generate varied labelled images, to expand the dataset while retaining crucial characteristics (Yang H. et. al. 2020).

A Pix2Pix GAN is used in this work for image augmentation. It is a type of conditional GAN that uses additional information to condition both the generator and the discriminator. This allows the generator to generate images related to additional information given. Hence, it will be able to generate realistic images in a smaller number of iterations. Through iterative training, the generator's output improves as the discriminator provides feedback on distinguishing real images from generated ones (Aljohani A. & Alharbe N., 2022).

The labelled images are used as ground truth images for conditioning the GAN. The Pix2Pix-style cGAN, depicted in Figure 5.2, involves training two models in opposition: the generator and the discriminator.

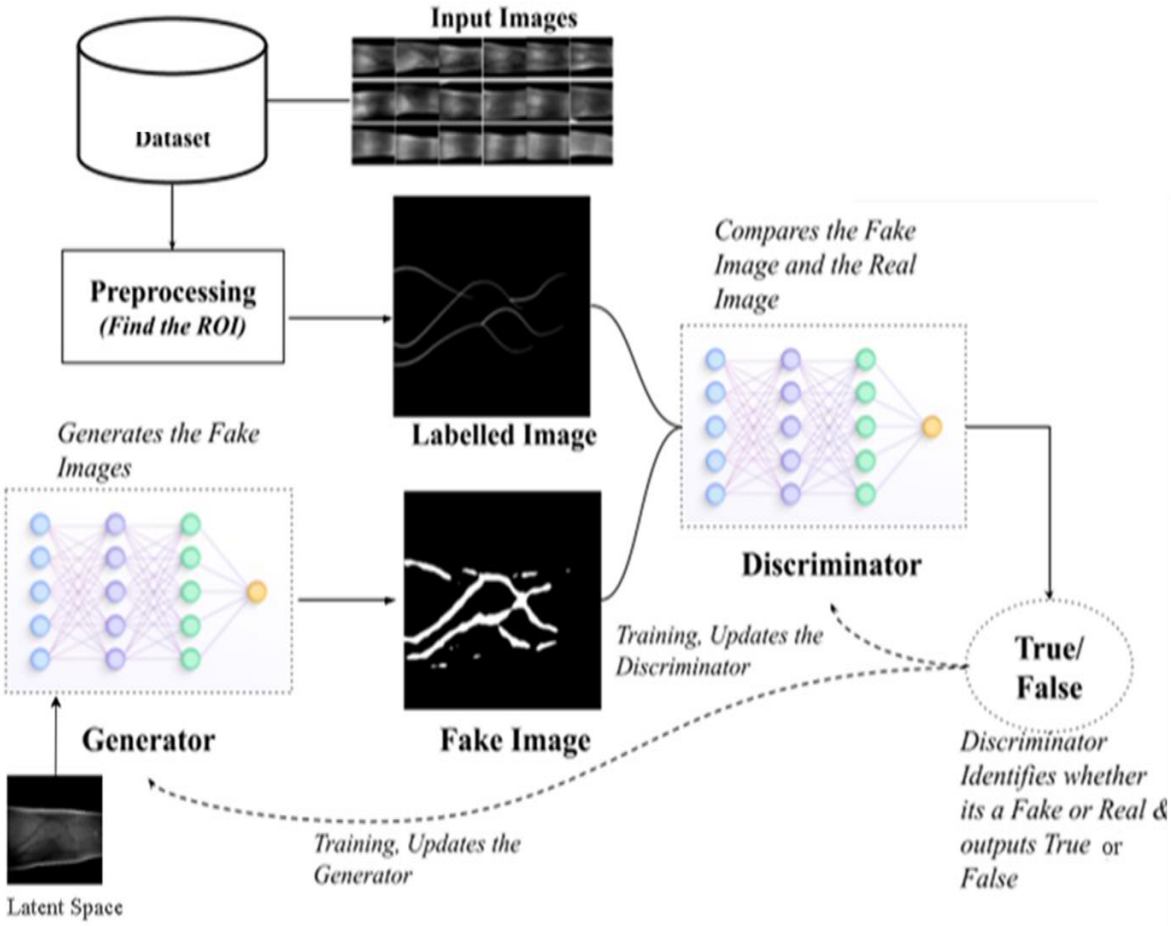


Figure 5.2 cGAN for Finger Vein Image Augmentation

The generator creates synthetic images using an autoencoder, with the tanh activation function defined by equation 5.4:

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \quad (5.4)$$

Tanh outputs values between -1 and 1, which compresses the output to a specific range, facilitating easier learning and reconstruction of the input data. Its zero-cantered nature also helps to alleviate the vanishing gradient problem, improving the model's convergence.

The discriminator employs a standard model with leaky ReLU and sigmoid activation functions. The leaky ReLU activation is given by the equation 5.6, where the function outputs x for positive values and a small value for negative inputs, allowing the network to learn from all inputs.

$$f(x) = \max(0.01 * x, x) \tag{5.5}$$

This activation ensures that the network can still learn from negative values. The discriminator classifies the images as either realistic or fake. The sigmoid activation function, shown in the equation 5.6, maps the discriminator's output to the range [0,1], where outputs closer to 1 indicate real images and those closer to 0 suggest fake images.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{5.6}$$

The discriminator receives two inputs: a labelled image and a fake image, which is generated by the generator. The fake image and the target image are compared by the discriminator and provide a true or false output. Training happens alternately for the generator and discriminator. The generator produces images similar to the target image. Discriminator compares the target and generated images. During the training of the discriminator, the generator remains fixed and vice versa. Finally, the GAN learns to generate images that resemble the target image.

In the equations from 5.8 to 5.11, x is the real data, and z is the latent vector. G(z) is the data generated by the generator. D(x) is the discriminator's assessment of genuine data. D(G(z)) is the discriminator's assessment of fake data.

The discriminator's parameters are updated using the gradient descent equation 5.7:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(x^{(i)}\right) + log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right)\right] \tag{5.7}$$

The generator is updated using the gradient descent shown in equation 5.8:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \tag{5.8}$$

The generator and discriminator have its unique loss function. The discriminator's loss function is expressed in the equation 5.9:

$$LD=Loss(D(x),1) + Loss(D(G(z)),0) \quad (5.9)$$

The generator's loss function is designed to make the discriminator as confused as possible. The generator's loss function is given by the equation 5.10:

$$LG = Loss(D(G(z)),1) \quad (5.10)$$

where x is the real data, z is the latent vector, G(z) is the fake data generated by the generator, D(x) is the discriminator's evaluation of real data, and D(G(z)) is the discriminator's evaluation of fake data.

These loss values indicate GAN's training progress and performance. They help evaluate the GAN's effectiveness in generating realistic images. The three loss values considered are:

- **d_loss1 (Discriminator Loss on Real Images)**: This indicates how accurately the model distinguishes real images from fakes. Generally, a lower value suggests better discrimination.
- **d_loss2 (Discriminator Loss on Fake Images)**: This measures how accurately the discriminator identifies fake images. A lower score indicates improved detection.
- **generator_loss (Generator Loss)**: This evaluates the generator's ability to create realistic images. This includes components like adversarial loss, promoting realism, and L1 loss, ensuring the output closely matches the target image.

Table 5.1 shows the loss values for d_loss1, d_loss2, and generator loss over various epochs of GAN training. The dynamic and fluctuating nature of the loss values is because of the adversarial relationship between the generator and the discriminator.

Table 5.1 Loss Values of cGAN for Different epochs

Epochs	d_loss1	d_loss2	Generator loss
1	0.3360	0.3610	5.6100
25	0.1930	0.2130	4.4360
50	0.2560	0.1450	4.7540
75	0.3530	0.3410	1.8093
100	0.2560	0.1450	2.3400
125	0.2560	0.2130	3.4360
150	0.1750	0.1450	3.5900
175	0.2450	0.2130	2.4360
200	0.2560	0.1450	1.7540
225	0.2482	0.1562	1.2953
250	0.2643	0.1847	1.3771
275	0.2561	0.2432	0.9839
300	0.1754	0.2012	1.2179

In the beginning, all loss values are relatively high, suggesting that the generator is struggling to produce real images. However, the discriminator is having difficulty distinguishing between real and fake images. The fluctuation in loss values during training indicates the competition between the generator and the discriminator. Each model constantly adapts and improves based on the performance of the other. By epoch 300, the discriminator's loss stabilizes, indicating that it has effectively learned to distinguish real from fake images. The generator also achieves the lowest loss, indicating that it has now the capability to generate more realistic images.

The Figure 5.3 shows the graph of the loss across various epochs during the training of the GAN.

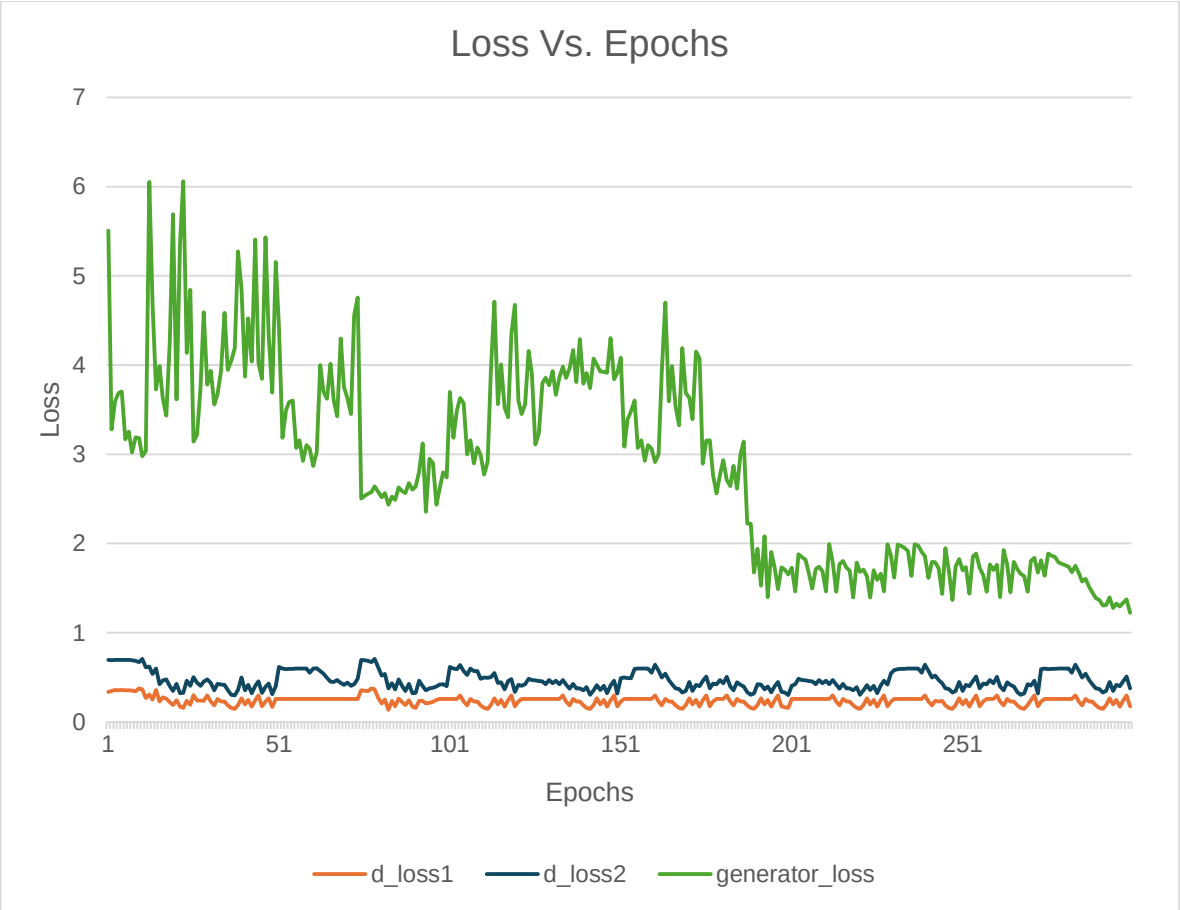


Figure 5.3. Generator and Discriminator Loss for 200 Epochs

The green line indicates the generator loss. The fluctuations in the early stages of training indicate its difficulty in generating realistic images. But gradually the loss decreases, indicating its improvement in creating realistic images.

The orange line indicates the discriminator loss on real images. The stable and low values of d_loss1 throughout the training process, that the discriminator performs well in distinguishing real images from generated ones.

The blue line shows the discriminator loss on fake images**.** Similar to d_loss1, d_loss2 also maintains a low and stable value. This indicates that the discriminator is consistent in recognizing generated images as fake.

As the training progresses, a consistent decrease in generator loss can be observed, suggesting that the generator is improving in its ability to produce realistic images. Meanwhile, the discriminator losses remain low, demonstrating its reliable capability in

differentiating between real and synthetic images. The interaction between these models is evident in the fluctuating generator loss, signifying the adversarial nature of GAN training, where improvements in one model led to challenges for the other. This graph effectively illustrates the progress and learning dynamics of the GAN over 200 epochs.

5.3 RESULTS AND DISCUSSIONS

5.3.1 Visual Analysis of Augmented Images

The images created using different traditional transformation techniques exhibit various orientations and lighting scenarios. Systematic rotations create images from different angles, shifts introduce spatial variability simulating changes in finger placement, and brightness variations reflect adaptability to different lighting conditions. Upon closer observation, the variation in the images is seen. Figure 5.4 shows a single sample original image and the augmented images created using conventional transformations. In Figure 5.5, more samples of images created using conventional transformations are displayed.

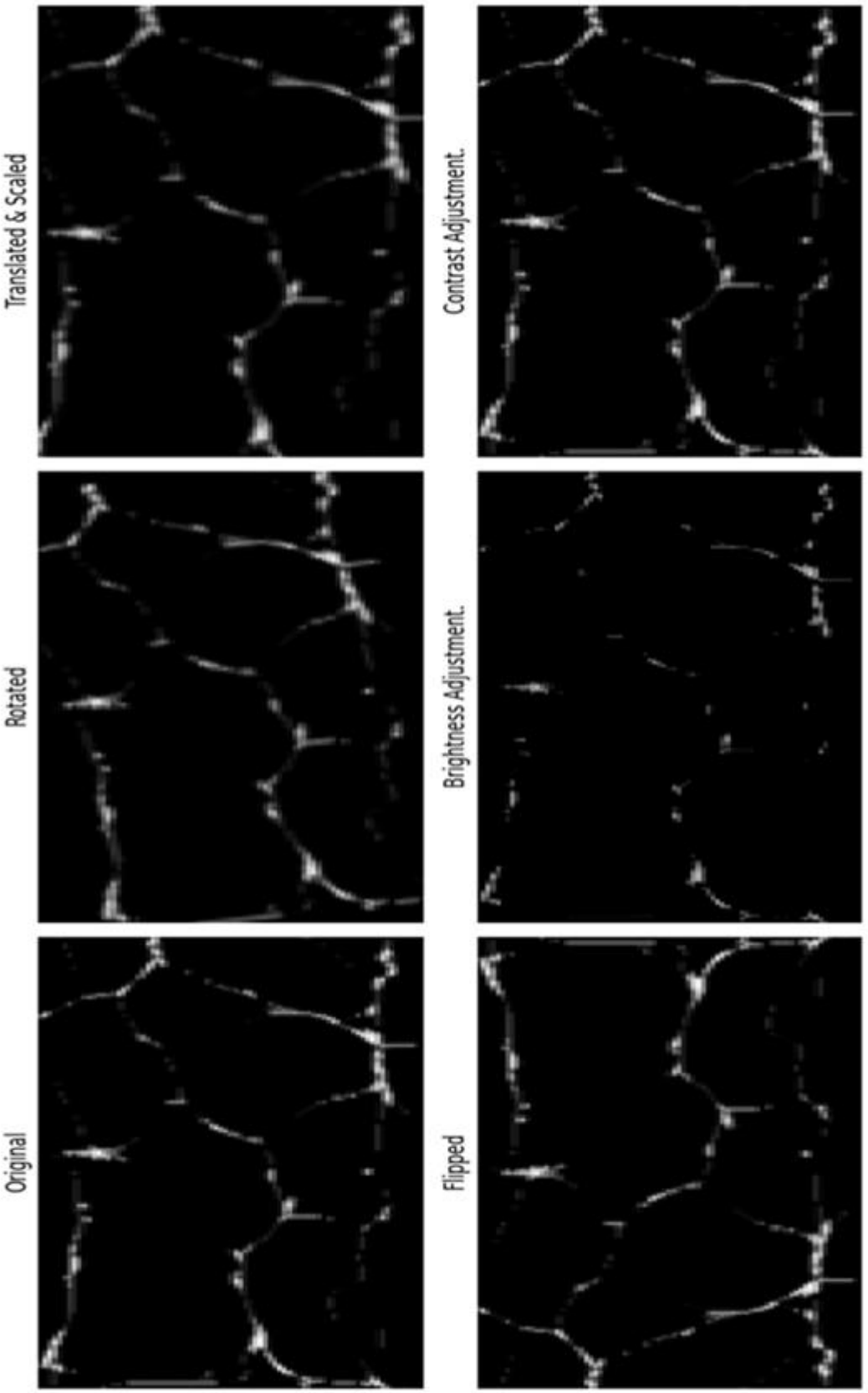


Figure 5.4 Original and Augmented Images Using Transformation Techniques

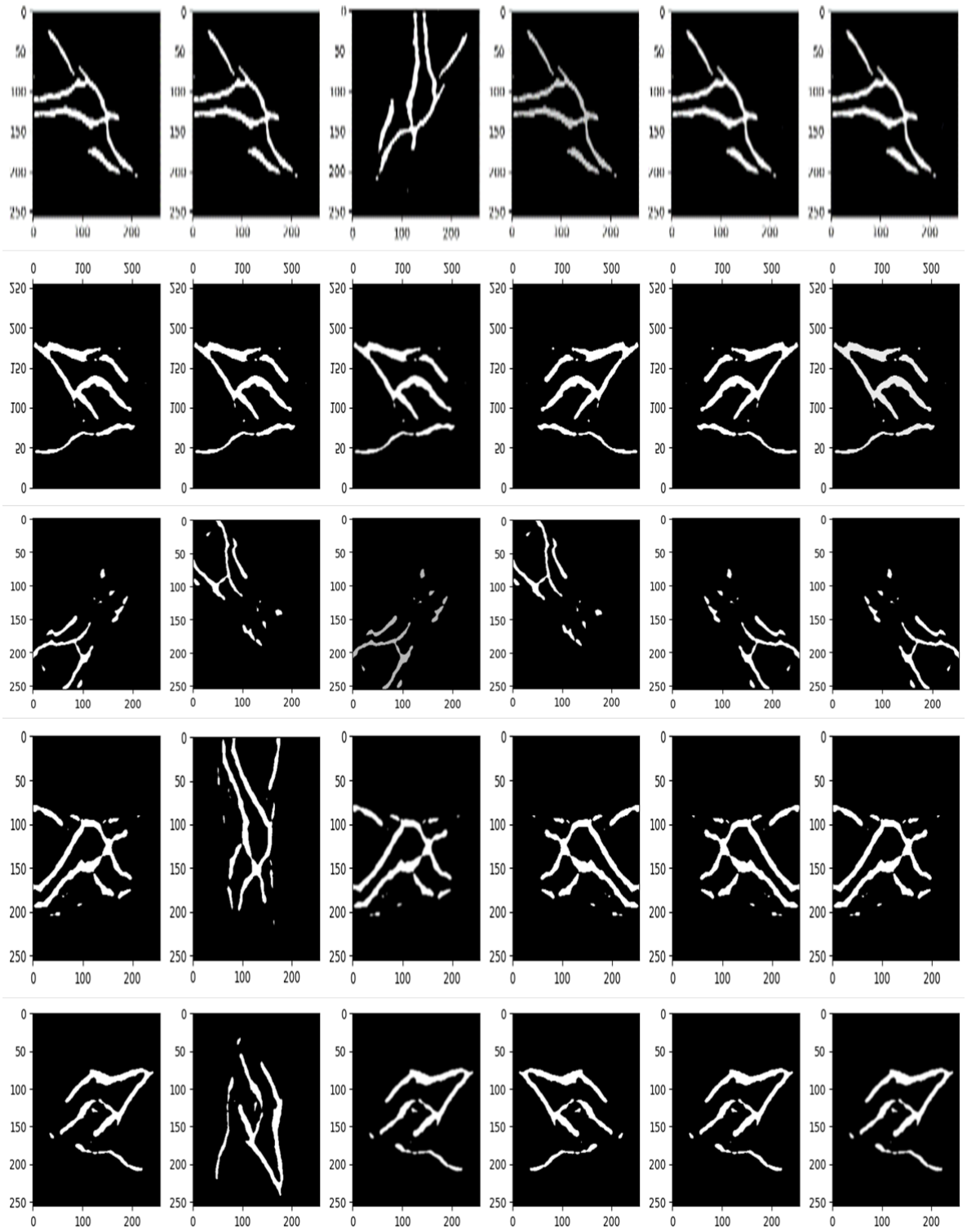
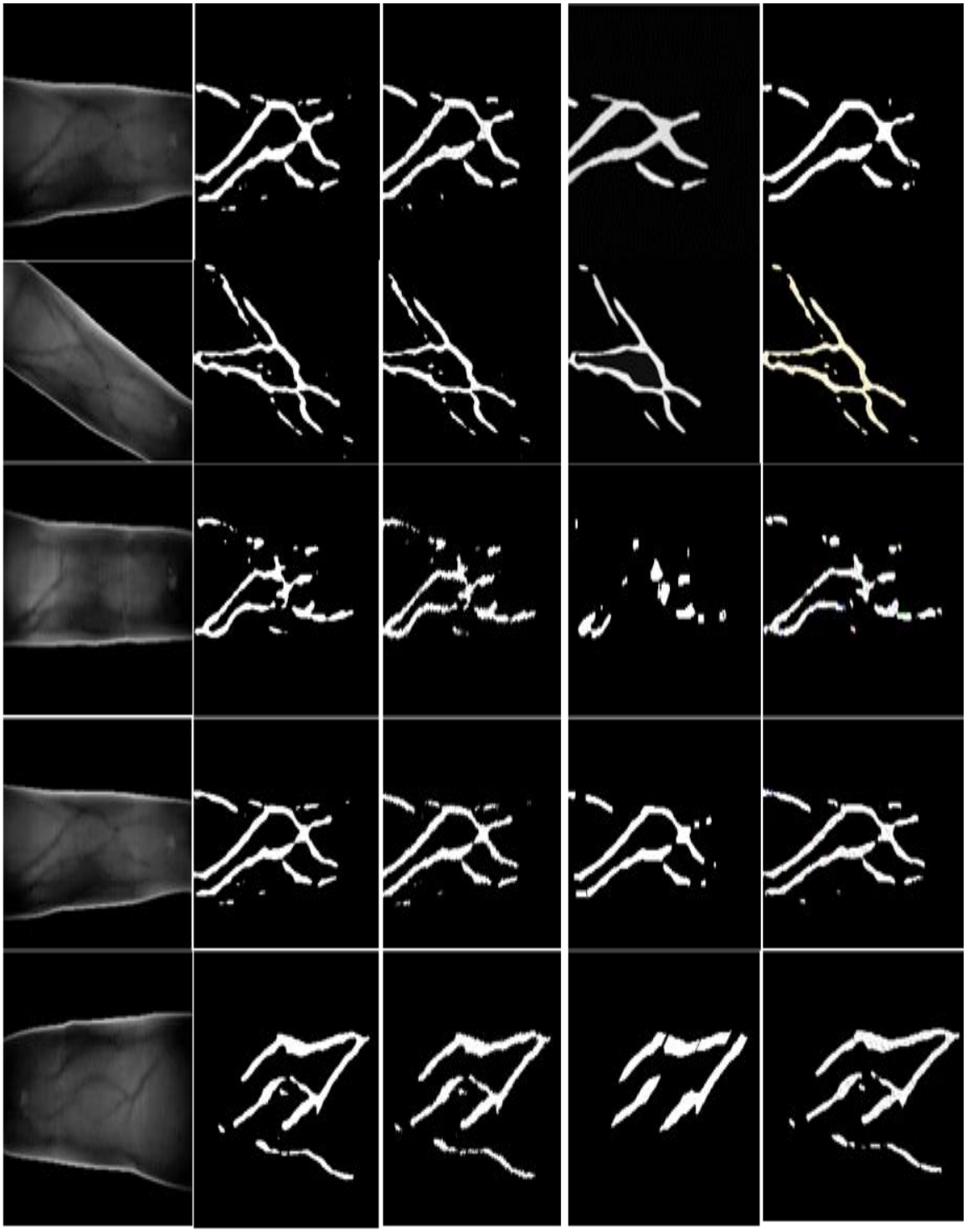


Figure 5.5. Sample Augmented Images using Conventional Transformations

Figure 5.6 showcases a set of sample images generated by the cGAN. These images are generated by the GAN during training, based on patterns it learned from real finger FV data.



(a) (b)

Figure 5.6 GAN Generated Images: (a) Input Image (b) Sample GAN Generated Images

Upon closer observation, variations and nuances within the generated images can be noticed. This shows the GAN's ability to introduce changes among the created images. These results show that the GAN not only captures the key characteristics of the training data but also introduces useful variations, thus helping in expanding the dataset. Some sample GAN-generated images simulating motion blur are shown in Figure 5.7.

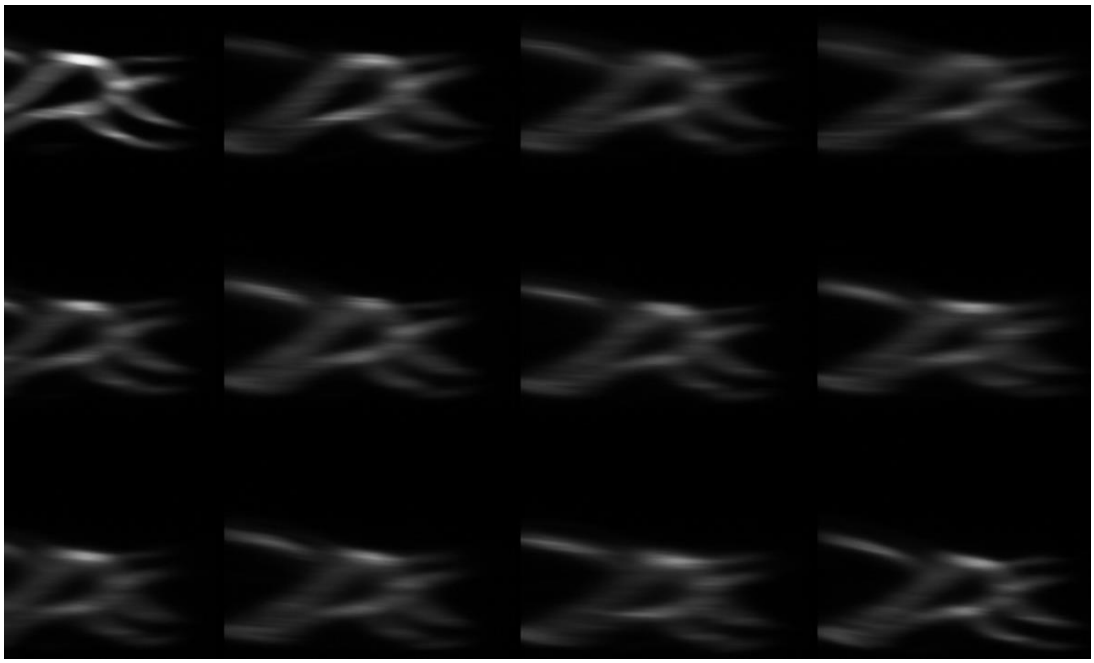


Figure 5.7 Images with Motion Artifacts Generated by cGAN

The successful deployment of a cGAN architecture has proved its adaptability to the domain of FVR, enabling the translation of source FV images into highly authentic representations. The adversarial training and the interplay between the generator and discriminator networks facilitated GAN to produce images that align with the unique characteristics of FV patterns.

5.3.2 Impact of Augmentation on Dataset Size

The dataset size has been considerably increased by including images created using conventional transformation techniques and GAN. Table 5.2 presents the details of images in the dataset before and after augmentation.

Table 5.2 Details of Images in the Dataset before and after Augmentation

Dataset	SDUMLA-HMT	THUFV
Original (Before augmentation)	106*6*6=3816	220*1= 220
Conventional Augmentation	5*6*106*10= 31800	220*10= 2200
GAN Augmentation	5*6*106*20= 63600	220*20= 4400
Testing set (without labelling) after Augmentation	6*106*30= 19080	1*220*5= 1100
Total Images	114480	7700
Training Set	80136	5610
Validation Set	15264	990

The SDUMLA-HMT dataset originally included 3,816 images collected from 106 individuals, with six fingers from each person captured six times. For testing purposes, one image per finger from each individual is set aside. The remaining five images per finger are first augmented using traditional techniques, generating ten new images for each original. GAN-based augmentation produced 20 more synthetic images per original image. Through this process, the total number of images in the dataset is increased to 80,136 images. Each class now contains 150 images, which are divided between the training and validation sets. The test images also go through augmentation, resulting in 30 versions per finger. After all forms of augmentation, the dataset is expanded with a total of 114,480 images.

The THUFV dataset consists of 220 images, with one image per individual. Conventional transformation methods increased the number of images to 2,200. Ten new images per original image are created during this process. Augmentation using GAN expanded the dataset to 4,400 images with 20 images per original image. One image per individual is reserved for testing. This is also augmented to create 5 images per individual, resulting in a total of 1,100 testing images. Thus, the THUFV dataset is expanded to contain 7,700 images. Among these, 5,610 images are allocated for training, while 990 images are used for validation.

5.3.3 Performance Evaluation of the Augmented Dataset on VGG16

VGG16 is used to evaluate the effectiveness of the augmented dataset. For the SDUMLA-HMT dataset, the original set of 3,816 images is increased to a total of 114,480 images. Similarly, the THUFV dataset is augmented to a total of 7,700 images. In both cases, the augmented images were used to train the VGG16 model by splitting the data for training, validation, and testing purposes. For testing, a combination of original and augmented images is used.

The distribution of the augmented dataset into training, validation, and testing sets is shown in Table 5.3.

Table 5.3 Distribution of Augmented Dataset for Training, Validation and Testing

Dataset	Training Dataset (%)	Validation Dataset (%)	Testing Dataset (%)
SDUMLA-HMT	70	13.3	16.7
THUFV	72.8	12.8	14.4

The images are divided based on each individual or class in the original dataset. This results in a fractional number for the percentage of each category. For the SDUMLA-HMT dataset, 70% of the images are allocated for training, 13.3% for validation, and 16.7% for testing. Similarly, for the THUFV dataset, 72.8% of the images are used for training, 12.8% for validation, and 14.4% for testing.

a. THUFV Dataset

Figure 5.8 shows the training and validation accuracy behaviour of the VGG16 model using a labelled and augmented dataset over 40 epochs.

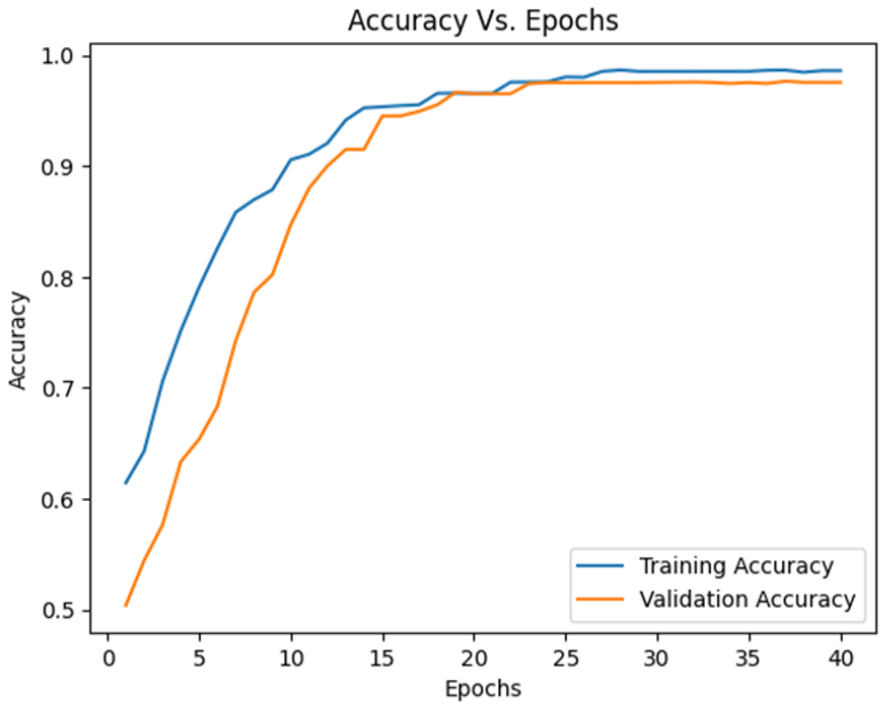


Figure 5.8 Accuracy Curve of VGG16 on Labelled and Augmented THUFV Dataset

The accuracy graph shows that the model is learning effectively. Both training and validation accuracy show constant improvement over the epochs, with no signs of overfitting. This consistent rise in validation performance implies that the model is generalizing well beyond the training data.

Figure 5.9 shows the training and validation loss behaviour of the VGG16 model using a labelled and augmented dataset over 40 epochs.

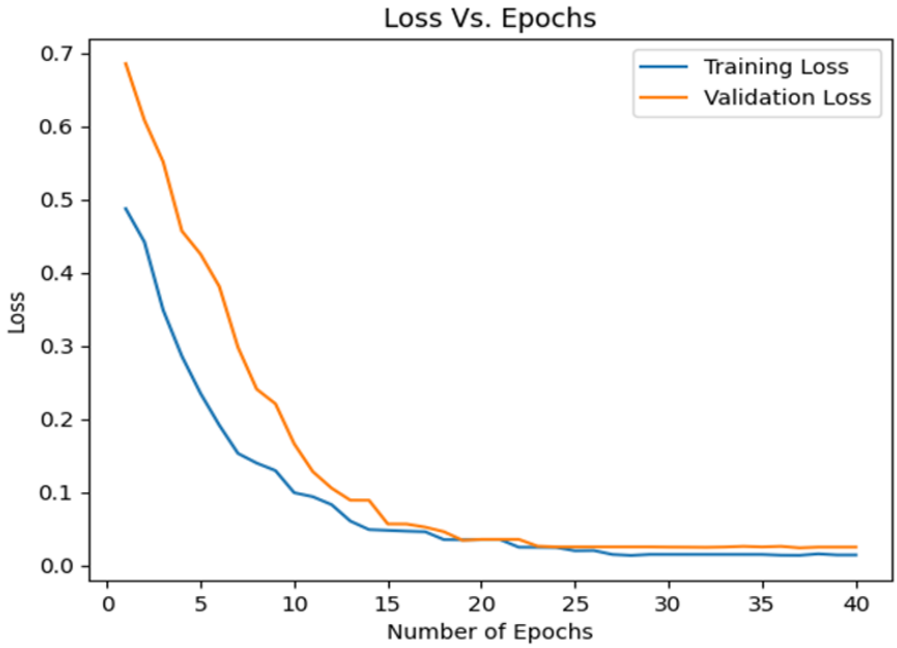


Figure 5.9 Loss Curve of VGG16 on Labelled and Augmented THUFV Dataset

The loss graph in Figure 5.9 shows a decreasing trend in both training and validation loss, gradually moving closer to zero. This indicates that the model is learning to reduce its prediction errors over time. The converging and stabilizing nature of the curves suggests that the model is fitting the training data and also generalizing to new data. Thus, overfitting has been mitigated, which is a good indication of balanced learning.

Table 5.4 demonstrates the performance improvements of the VGG16 model with the application of labelling and augmentation techniques. The table contains the performance details of VGG16 with the original dataset, VGG16 trained over the labelled dataset (L-VGG16) and VGG16 trained over the labelled and augmented dataset (LA-VGG16).

Table 5.4 Performance Comparison of VGG16 on Original Dataset, Labelled Dataset and Labelled-Augmented THUFV dataset

Configuration	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
VGG16	92.5	94.09	97.64	95.83
L-VGG16	96.34	98.29	97.87	98.08
LA-VGG16	98.6	99.6	98.97	99.29

With the original dataset VGG16 model achieves an accuracy of 92.5%, a precision of 94.09%, a recall of 97.64%, and an F1-score of 95.83%. When labelling is introduced, the model's training accuracy improves by 3.84 %, reaching 96.34%. Precision increases by 4.2%, recall shows a slight improvement of 0.23%, and the F1 score rises by 2.25%. When both labelling and augmentation are applied, the training accuracy increases by 6.1%, precision increases significantly by 5.51%, recall improves by 1.33 %, and the F1-score increases by 3.46%. These results clearly illustrate that labelling and augmentation substantially boost the VGG16 model's performance across all metrics.

b. SDUMLA Dataset

The VGG16 performance is also analyzed using the SDUMLA dataset. The accuracy behaviour in Figure 5.10 shows that both training and validation accuracy increase and both accuracies converge gradually, indicating the model's generalization capability.

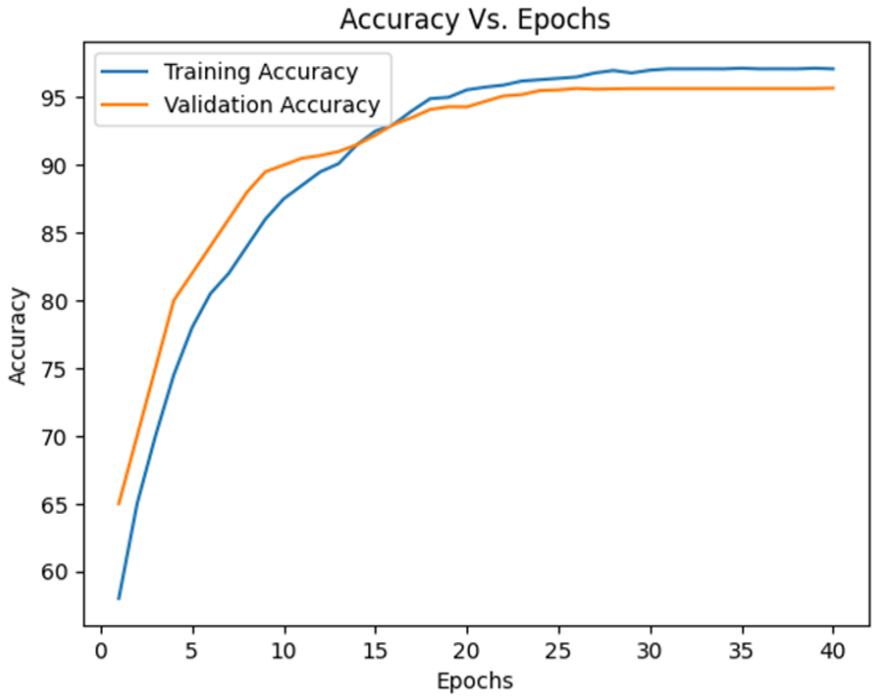


Figure 5.10 Accuracy Curve of VGG16 on Labelled and Augmented SDUMLA Dataset

Figure 5.11 highlights the reduction in both training and validation loss, with both metrics showing a sharp decline initially and then stabilizing as the epochs progress.

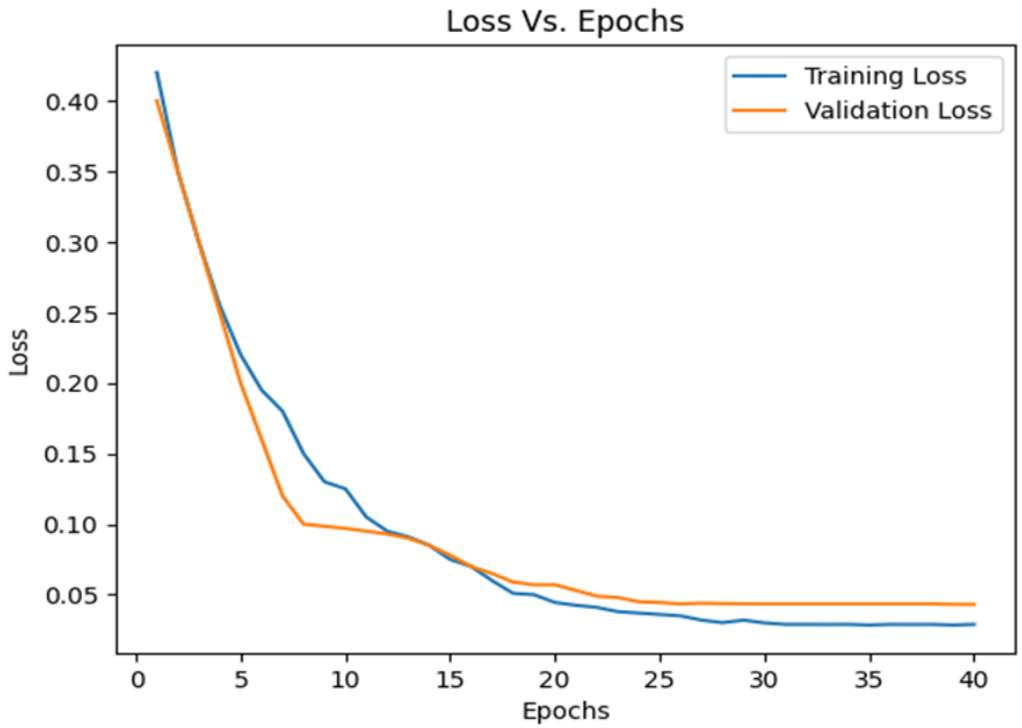


Figure 5.11 Loss Curve of VGG16 on Labelled and Augmented SDUMLA Dataset

The loss curve also decreases gradually and converges, suggesting that the model is learning effectively without overfitting. Table 5.5 provides a comparison of the performance of VGG16 on the SDUMLA dataset when trained on the original data, labelled data, and labelled-augmented data.

Table 5.5 Performance Comparison of VGG16 on Original Dataset, Labelled Dataset and Labelled-Augmented SDUMLA Dataset

Configuration	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
VGG16	94.31	93.60	92.07	92.83
L-VGG16	95.45	94.78	96.76	95.76
LA-VGG16	97.10	95.73	97.11	96.41

Initially, the VGG16 model achieves a training accuracy of 94.31%, with precision, recall, and F1 score at 93.60 %, 92.07 %, and 92.83 %, respectively. After labelling the data, the training accuracy increased to 95.45 %, precision to 94.78 %, recall to 96.76 %, and F1-score to 95.76%. This recall and F1-score improved significantly. Data augmentation leads to even better results, with a notable increase of 2.79% in training accuracy, 5.04% in recall, and 3.58% in F1-score. These enhancements demonstrate the impact of data labelling and augmentation on the performance of the model.

5.4 SUMMARY

In this chapter, several dataset augmentation strategies and their impact on model performance are investigated. Classic image transformations like rotation, shifting, zooming, illumination variation etc and GAN-based augmentation are used for increasing the dataset size, and the results of both approaches are detailed. The performance of the augmented datasets is evaluated using VGG16. The experimental results show that augmentation improves the performance of the VGG16 model across all the metrics evaluated.