Hairlines removal and low contrast enhancement of melanoma skin images using CNN with aggregation of contextual information

Report submitted to the SASTRA Deemed to be University as the requirement for the course

ICT300 - MINI PROJECT

Submitted By

(Reg.No:124014010, B.Tech-ICT)
Geetika Chandrashekhar
(Reg.No:124014020, B.Tech-ICT)
Karmukilan V
(Reg.No:124014042, B.Tech-ICT)
Sanchana S

SCHOOL OF COMPUTING

THANJAVUR, TAMIL NADU, INDIA -613401

SCHOOL OF COMPUTING

THANJAVUR –613401

Bonafide Certificate

This is to certify that the report titled "**Hairlines removal and low contrast enhancement of melanoma skin images using CNN with aggregation of contextual information**" submitted as a requirement for the course, ICT300: **MINI PROJECT** for B.Tech. is a bonafide record of the work done by **Geetika C (Reg.No:124014010, B.Tech-ICT) , Karmukilan V (Reg.No:124014020, B.Tech-ICT) , Sanchana S (Reg.No:124014042, B.Tech-ICT)** during the academic year 2022-23, in the School of Computing, under my supervision.

Signature of Project Supervisor :

Name with Affiliation : Joy Christy A, Assistant Professor –II , Dept of soc

Date : 18/05/2023

Mini Project *Viva voce* held on 18/05/2023.

Examiner 1 **Examiner 2**

Acknowledgements

We would like to thank our Honorable Chancellor **Prof. R. Sethuraman** for providing us with an opportunity and the necessary infrastructure for carrying out this project as a part of our curriculum.

We would like to thank our Honorable Vice-Chancellor **Dr. S. Vaidhyasubramaniam** and **Dr. S. Swaminathan,** Dean**,** Planning & Development, for the encouragement and strategic support at every step of our college life.

We extend our sincere thanks to **Dr. R. Chandramouli,** Registrar, SASTRA Deemed to be University for providing the opportunity to pursue this project.

We extend our heartfelt thanks to **Dr. A. Umamakeswari**, Dean, School of Computing, **Dr. B.Santhi**, Associate Dean, Department of Computer Application, **Dr. R. Muthaiah**, Associate Dean, Department of Information Technology and Information & Communication Technology and **Dr. V. S. Shankar Sriram**, Associate Dean, Department of Computer Science and Engineering for their motivation and support offered in materializing this project.

Our guide **Joy Christy A , Assistant Professor –II** , School of Computing was the driving force behind this whole idea from the start. Her deep insight in the field and invaluable suggestions helped us in making progress throughout our project work. We also thank the project review panel members for their valuable comments and insights which made this project better.

We would like to extend our gratitude to all the teaching and non-teaching faculties of the School of Computing who have either directly or indirectly helped us in the completion of the project.

We gratefully acknowledge all the contributions and encouragement from my family and friends resulting in the successful completion of this project. We thank you all for providing me an opportunity to showcase my skill through project.

List of Figures

List of Tables

Abstract

The proposed approach intends to increase the precision of skin lesion photographs that medical professionals use to identify and treat melanoma, a particular type of skin cancer. The presence of the hairline, which could cover up the lesions and make them hard to recognise, is one of the difficulties when analysing these images. The suggested approach addresses this issue by first preprocessing the image to get rid of hairlines. The image's contrast is then improved with the use of a (CNN) model.

The suggested technique makes use of a CNN model that has been trained to take into account contextual information and connections between nearby pixels in images. This enables the model to recognise lesions and isolate them from surrounding skin with greater accuracy. In order to distinguish between local and global features, CNN models are made to incorporate contextual data from several image sizes. In general, the suggested method might enhance skin lesion imaging, resulting in more precise melanoma detection and therapy. The performance of the suggested method was assessed using the 2018 ISIC Challenge dataset, which includes a number of images of skin lesions. The data set was turned into a training set, and the CNN model developed using the test set. We evaluated how well the proposed strategy for erasing hairlines and enhancing contrast loss functioned in comparison to existing approaches. The evaluation's findings demonstrate that the suggested method works better in low-contrast enhancement and hairline removal than the rival methods. To ascertain how each element of the recommended technique participated the authors also carried out an ablation research. The ablation study's findings proved that for exceptional results, both the application of contextual knowledge and hairline reduction are required.

KEY WORDS: Low contrast enhancement, melanoma skin pictures, convolutional neural network, contextual information, data augmentation, and blackhat algorithm Peak signal to noise

ratio, structural similarity index measures, and Natural picture quality evaluator, medical image processing, and mean square error.

Table of Contents

CHAPTER 1

SUMMARY OF THE BASE PAPER

1.1 Base Paper Details:

Title : Hairlines removal and low contrast enhancement of melanoma skin images using CNN with aggregation of contextual information

Journal : Biomedical Signal Processing and Control

Author :Ranpreet Kaur , Hamid GholamHosseini , Roopak Sinha .

Publisher : Received 29 December 2021, Revised 3 March 2022, Accepted 13 March 2022, Available online 19 March 2022, Version of Record 19 March 2022.2666-1659/© 2020 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY-NC-ND license.

1.2 Summary:

The goal of this project is to improve the detection of melanoma skin cancer using convolutional neural networks (CNNs). Researchers proposed preprocessing strategies for melanoma skin scans that included hairline removal and contrast enhancement techniques. These preprocessed images are input to the CNN to determine whether they are malignant or benign. The researchers also used a patch-based methodology to integrate clusters of contextual data to improve CNN accuracy. This technique allows CNNs to consider the whole picture rather than just specific pixels, potentially improving their ability to identify small, irregular melanoma tumours. In general, this study found that the proposed strategy is superior to existing methods in accurately classifying melanoma skin images.

1.3 Introduction:

Melanoma skin cancer can be fatal if not recognized and treated promptly. Early detection is essential for the treatment of melanoma, as accurate diagnosis mainly depends on medical imaging analysis. The presence of hairlines, low contrast, and other visual features that can interfere with the analysis process make it difficult to reliably identify melanoma within skin images. To solve this problem, convolutional neural networks (CNNs) are used in new preprocessing techniques for melanoma skin images. To improve the quality of the photos for input to the CNN, it has been proposed to remove the hairline and enhance the low contrast of the skin photos. The performance of CNNs in detecting small, atypical melanoma tumours was further improved by the researchers' use of contextual data ensembles. The proposed method shows promising results in the early detection of melanoma using skin scans and may aid physicians in early detection and diagnosis of melanoma skin cancer. This effort could make medical image analysis more accurate, ultimately helping patients.

1.4 Blackhat Algorithm:

Morphological black hat techniques in image processing can be used to remove hairlines from images of the skin or scalp. Hairlines can be difficult to understand in skin images, especially in medical imaging where they may prevent the accurate detection of skin illnesses or disorders. Morphological black hat surgery can help remove hairlines from images by increasing the contrast of the skin against the hair and background.

The black hat operation increases the contrast of the skin while removing noise from the hair and background by measuring the difference between the input image and its morphological closing. The outcome of the black hat procedure is a photograph that minimises the visibility of hairlines while enhancing the contours of the skin.
Because it enhances the quality of the images for input into image analysis algorithms, the black hat operation for hairline removal can be helpful in a variety of applications, including the analysis of skin lesions and scalp problems. As with any image processing technology, it is important to take into account the individual application and any potential process restrictions in order to produce findings that are accurate and trustworthy.

1.5 MCACNN:

A patch of an image with variable dilation is subjected to 32 convolutions of size 3 3 in the MRAB block's convolutional approach. Each convolution layer's dilation rate is given in Convi.... ConvN, where N = 8, are 2, 4, 6, 8, 16, 32, 64, and 128. As the convolution layer of the network is utilised more frequently, it rises. The network employs dilation convolutions without intending to degrade the image's quality in order to collect the contextual information.
The adaptive normalisation layers change the strength of the branches for batch normalisation (BN), whereas the adaptive normalisation layers change the strength of the branches for identity normalisation (IBN). These layers provide batch normalisation and identity normalisation by back propagating the derivative of the loss function. In order to filter data computed by the preceding layer that travels forward via the network, the output of both the and scaling layers are pooled and given to the activation function leaky ReLU.
The output is finally rebuilt using a convolutional operation that employs 32 filters of size 1 1 and a regression layer. The loss function used by the regression layers to assess the learnability of the aggregation procedure is the mean square error (MSE) function. This loss function between the target picture and the network's predicted image is used to calculate the mean square error for R replies.

1.6 Bilateral filtering:

Bilateral filtering is a typical non-linear filtering technique to smooth images while keeping edges. The method uses a weighted average of the neighbouring pixels in an image, where the weights depend on how similar the pixels are to one another in terms of both intensity and spatial proximity. Bilateral filtering involves applying a Gaussian filter to the spatial and intensity domains of a picture. The spatial filter regulates the neighbourhood range, which establishes how far away from the centre pixel surrounding pixels will be included during the averaging process. The intensity filter controls the similarity range, which regulates how different the intensity values between the central and surrounding pixels can be before the weight is 0.

The bilateral filter is helpful in a variety of image processing applications, including image denoising, picture smoothing, and feature recognition because it is very successful in eliminating noise from images while preserving edges. It should only be used sparingly though, as it could result in abrupt shifts to the image or blurring in areas of strong contrast.

Overall, bilateral filtering is a useful tool in the arsenal of image processing techniques that may be tailored and improved for certain applications. It is a popular option in a variety of image processing domains, including computer vision, medical imaging, and digital photography, because of its effectiveness and efficiency.

1.7 Clahe:

CLAHE (Contrast Limited Adaptive Histogram Equalisation), a technique used in image processing, boosts contrast in an image while keeping its dynamic range. It is a variation of the histogram equalisation method, which is widely used to change the intensity distribution of an image so that a more uniform histogram is produced.

In CLAHE, the histogram equalisation process is applied to each tile separately using the histogram equalisation approach. With this approach, it is possible to prevent the potential overamplification of noise in low-contrast areas that could occur with traditional histogram equalisation.

1.8 Performance Metrics:

1.8.1 PSNR:

Peak Signal-to-noise ratio, or PSNR, is a statistic used in the processing of pictures and videos to determine the way a clip or image has been compressed or rebuilt compared to its original source. The original and compressed/reconstructed versions are compared using the percentage of signal to noise (PSNR), which is the ratio of a signal's highest possible strength to the amount of noise that affects the accuracy of its representation.

Because PSNR offers a value that indicates the degree of distortion or loss in the image or video as a result of compression or other processing, it is frequently employed as a quantitative indicator of image and video quality.

The PSNR number increases with the amount of distortion or loss in the image or video. The mean square error (MSE) between the original and the compressed/reconstructed version is divided by the maximum pixel value, and this ratio is used to calculate PSNR.

$$PSNR = 20 * \log10(MAX) - 10 * \log10(MSE) \tag{1}$$

where MAX is the largest pixel value that may be displayed (usually 255 for an 8-bit image) and MSE is the mean square error between the original and the compressed/reconstructed version.
Since PSNR does not account for the perceptual variations in human vision and may not fully reflect the subjective visual quality of an image or video, it has significant limitations as a measure of the quality of images and videos. In the realm of photo and video processing, it is still a popular and helpful statistic, especially when a quantitative comparison between the original and the compressed/reconstructed form is required.

1.8.2 MSE:

Two sets of data are compared using this statistical technique. The habit of comparing the original image with the compressed or rebuilt image is prevalent in image processing.
Since there is less variance between the original and reconstructed or compressed images, a lower MSE score is regarded to be a more accurate depiction of the original image. The visual quality of an image may not always be accurately reflected by MSE because it cannot take into account the variances in how people see things. MSE is a well-liked and helpful metric in the realm of image processing when assessing the effectiveness of image and video compression methods or comparing the standards of various image processing approaches.
Each pixel value in the original image is squared to determine the MSE.

The equation for calculating MSE is as follows:

$$MSE = (1/N) * \sum [i=1 \text{ to } N] (I(i) - K(i))^2 \quad (2)$$

where I(i) is the pixel value from the original image at position i, K(i) is the matching pixel value from the compressed or reconstructed image at position i, N is the total number of pixels in the image.

1.8.3 SSIM:

A measure is used to judge how similar two images are. It is a popular and useful statistic for evaluating the efficacy of methods for enhancing, restoring, and compressing images in the field of image processing. The structural information and brightness information of the images are both taken into consideration by SSIM, unlike the Mean Square Error (MSE) measure. This demonstrates that SSIM is more effective at capturing changes in perceived human vision than MSE.

The three pillars on which SSIM compares the similarity of two images are brightness, contrast, and structural information. The SSIM index value might be anything between -1 and 1, with 1 denoting a full mismatch between the two images.

The following equation is used to determine the SSIM index:

$$SSIM (x, y) \text{ is equal to } [l(x, y) * c(x, y) * s(x, y)]. \quad (3)$$

The brightness, contrast, and structural similarity measurements are l(x,y), c(x,y), and s(x,y), respectively, at each pixel (x,y) and are constants that control the relative importance of each component. The two images being compared are x and y.

Contrast describes the variations in intensity between neighbouring pixels, whereas luminance represents the overall brightness or intensity of the image. SSIM provides a more accurate measurement of the similarity between two images than traditional measures like MSE, especially when the images have complex patterns and textures. When evaluating the efficacy of image processing algorithms, it is still essential to take SSIM into consideration in addition to subjective visual quality assessment.

1.8.4 NIQE:

The statistical aspects of a picture, such as its power spectrum, the distribution of its image patches, and the mean and variance of its gradients, are all examined by NIQE to assess its quality. The method is based on the notion that natural photographs have distinctive statistical properties that distinguish them from photos created artificially, such as noise or blur. Recall that

NIQE is a no-reference measure, which means that it does not require a reference image to evaluate image quality. Instead, it just considers the image's statistical properties while making a judgement.
Numerous applications have shown how successful NIQE is, including picture and video compression, image restoration, and image enhancement. The NIQE score for a particular image is determined using a vector of statistical information that the trained model produces. Following the transformation and fusion of these features using a set of weights, the image quality is then represented by a single scalar score. The actual algorithm used to calculate this score is private and not made available to the general public because it depends on the exact architecture and parameters of the trained model.

Table 1.1 Results of the performance metrics of each class

	CLAHE				BILATERAL				MCACNN			
DISEASES	PSNR	MSE	SSIM	NIQE	PSNR	MSE	SSIM	NIQE	PSNR	MSE	SSIM	NIQE
AK	29.303	76.916	0.834	8.753	40.942	5.559	0.9362	8.753	42.693	3.741	0.956	9.481
BCC	29.679	71.280	0.841	17.497	41.243	5.638	0.9337	8.308	42.814	4.062	0.951	8.902
BKL	29.668	71.087	0.832	19.359	40.186	6.692	0.9211	9.372	41.465	5.090	0.938	9.787
DF	29.440	74.905	0.82	19.500	39.706	7.540	0.909	8.677	40.870	5.956	0.927	9.272
MEL	29.657	71.373	0.842	20.065	40.002	6.994	0.921	11.325	41.253	5.321	0.939	11.219
NV	30.240	62.524	0.842	18.785	40.632	6.121	0.924	8.806	41.852	4.726	0.940	9.147
VASC	30.326	61.88	0.843	19.485	40.806	5.891	0.927	9.666	41.921	4.712	0.940	10.004

Table 1.2 Average results of the performance metrics

Method	PSNR	MSE	SSIM	NQE
CLAHE	29.759	69.995	0.836	17.634
BILATERAL FILTERING	40.502	8.381	0.924	9.272
MCACNN	41.838	4.801	0.941	9.210

Dataset:

Dataset Name: HAM10000 (Source: Kaggle)

Description: It held 16,838 dermoscopic training pictures gathered from various patients. As shown in the table, many diagnostic classifications of lesions are present, including dermatofibroma (DF), basal cell carcinoma (BCC), benign keratoses (BKL), melanoma (MEL), melanocytic nevi (NV), and vascular lesions (VASC)

No. of samples	AK	BCC	BKL	DF	MEL	NV	VASC	Total
Training	867	3323	2624	239	4522	5010	253	16838
Test	–	–	–	–	–	–	–	8239

CHAPTER 2

MERITS AND DEMERITS OF THE BASE PAPER

Merits:

In order to enhance the visual quality of melanoma skin images, the publication "Hairlines removal and low contrast enhancement of melanoma skin images using CNN with aggregation of contextual information" suggests a novel method. The following are some advantages of this strategy:

1. Better image quality: The suggested method can successfully eliminate hairlines from photos of melanoma skin and boost the low contrast regions, producing images with higher visual quality.

2. Contextual information aggregation: The suggested method aggregates contextual information from nearby pixels using a convolutional neural network (CNN) to improve contrast and get rid of hairlines. As a result, the method is more resistant to noise and changes in lighting.

3. comparatively low computational complexity: When compared to existing image enhancing methods, the suggested method has a comparatively low computational complexity, which makes it more useful for real-time applications.

4. Versatility: The suggested method is a versatile approach for enhancing skin photographs since it may be used on a variety of skin images with different lighting and hairline patterns.

5. Objective evaluation: The suggested method's performance is assessed using objective metrics such as the Peak Signal-to-Noise Ratio (PSNR) and Structural Similarity Index (SSIM), which offer a quantifiable indicator of the enhancement of image quality.

Demerits:

In spite of the fact that the publication "Hairlines removal and low contrast enhancement of melanoma skin images using CNN with aggregation of contextual information" has several advantages, there are a few potential downsides to take into account, including:

1. Dataset restrictions: In order to train the CNN model, the suggested strategy depends on having access to a large enough pool of training images. The effectiveness of the procedure may suffer if the dataset is biased or of restricted size.

2. Generalizability: Although the suggested method has produced encouraging results on skin melanoma photos, its applicability to other sorts of images, such as skin lesions that aren't melanoma, is still unclear.

3. Interpretability: It might be difficult to read the outcomes and comprehend how a deep learning system, like CNN, is generating decisions.

4. Computing requirements: Although the suggested method requires less computing power than some alternative image enhancement methods, it still needs a lot of computing power to train the CNN model.

5. Evaluation restrictions: While objective measurements like PSNR and SSIM offer quantifiable measures of the improvement in image quality, they could fall short in capturing the subjective perceived quality of the images, which is dependent on the preferences of the viewer.

CHAPTER 3
SOURCE CODE

#BLACKHAT ALGORITHM

```
import cv2
import os
folder_path = "C:/Users/Karmukilan/Pictures/skinnew/test/nv-p/"

def hairline_removal(img):
    grayScale = cv2.cvtColor( img, cv2.COLOR_RGB2GRAY )
    kernel = cv2.getStructuringElement(1,(17,17))
    blackhat = cv2.morphologyEx(grayScale, cv2.MORPH_BLACKHAT, kernel)
    ret,thresh2 = cv2.threshold(blackhat,10,255,cv2.THRESH_BINARY)
    dst = cv2.inpaint(img,thresh2,1,cv2.INPAINT_TELEA)
    return dst

for filename,i in zip(os.listdir(folder_path), range(len(os.listdir(folder_path)))):
    img= cv2.imread(os.path.join(folder_path,filename))
    output =hairline_removal(img)
    cv2.imwrite("C:/Users/Karmukilan/Pictures/skinnew/test/nv_op/"+str(i+1)+'.jpg', output)
```

#BILATERAL TRANSFORMATION

```
import cv2
import os
# Define the folder path containing the images
folder_path = "C:/Users/Karmukilan/Pictures/skinnew/test/nv_op/"

# Loop through each image in the folder
for filename in os.listdir(folder_path):
    if filename.endswith('.jpg') or filename.endswith('.png'):
        # Read the image
        img_path = os.path.join(folder_path, filename)
        img = cv2.imread(img_path)

        # Apply bilateral filtering
        filtered_img = cv2.bilateralFilter(img, 9, 75, 75)
```

```
        # Save the filtered image
        output_path = os.path.join(folder_path, f"filtered_{filename}")
        cv2.imwrite(output_path, filtered_img)
```

#CLAHE TRANSFORMATION

```
import cv2
import numpy as np
import os

# Define CLAHE function
def apply_clahe(img):
    # Convert image to LAB color space
    lab = cv2.cvtColor(img, cv2.COLOR_BGR2LAB)
    # Split channels
    l, a, b = cv2.split(lab)
    # Apply CLAHE to L channel
    clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
    cl = clahe.apply(l)
    # Merge channels
    merged = cv2.merge((cl,a,b))
    # Convert back to RGB color space
    result = cv2.cvtColor(merged, cv2.COLOR_LAB2BGR)
    return result

# Define image directory
img_dir = "C:/Users/Karmukilan/Pictures/skinnew/test/nv_op"

# Loop through images in directory and apply CLAHE
for filename in os.listdir(img_dir):
    if filename.endswith(".jpg") or filename.endswith(".png"):
        # Read image
        img = cv2.imread(os.path.join(img_dir, filename))
        # Apply CLAHE
        clahe_img = apply_clahe(img)
        # Save image
        cv2.imwrite(os.path.join(img_dir, "clahe_" + filename), clahe_img)
```

#SSIM

```
from skimage.metrics import structural_similarity as ssim #clahe
import cv2
import os
# Path to the directory containing the images
```

```
img_dir = "C:/Users/Karmukilan/Pictures/skinnew/test/nv_op/"
output_dir= "C:/Users/Karmukilan/Pictures/skinnew/test/nv_median/"

# List the files in the directory
img_files = os.listdir(img_dir)
output_files=os.listdir(output_dir)

imagepath = []
for img_file in img_files:
    imagepath.append(os.path.join(img_dir, img_file))
    #print(img_file)
outputpath = []
for img_file in output_files:
    outputpath.append(os.path.join(output_dir, img_file))
    #print(img_file)
scr=[]
for i in range(len(outputpath)):
    img = cv2.imread(imagepath[i])
    output = cv2.imread(outputpath[i])
    grayimg = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    grayout = cv2.cvtColor(output, cv2.COLOR_BGR2GRAY)
    # Compute the SSIM index between the original image and the grayscale image
    (score, diff) = ssim(grayimg, grayout, full=True)
    #print("SSIM score for", i,":", score)
    scr.append(score)
sum(scr)/len(scr)
```

#Peak signal to noise ratio

```
import numpy as np

mse_total = 0
psnr_total = 0
num_images = 0

diff_c=[]
for i in range(len(outputpath)):
    img = cv2.imread(imagepath[i])
    output = cv2.imread(outputpath[i])
    grayimg = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    grayout = cv2.cvtColor(output, cv2.COLOR_BGR2GRAY)

    diff = cv2.subtract(grayimg,grayout)
```

```
    #Set negative values to zero
    diff[diff < 0] = 0

#Calculate the mean squared error (MSE)
    mse = np.mean((grayimg - grayout) ** 2)
    mse_total += mse
    #print(mse_total)

 # Calculate the maximum pixel value of the images
    max_pixel = 255.0

 # Calculate the PSNR using the formula
    psnr = 10 * np.log10(np.square(max_pixel) /mse)
    psnr_total += psnr
    #print(psnr_total)

 # Increment the count of images processed
    num_images+= 1

# Calculate the average PSNR over all images
avg_psnr = psnr_total /num_images
avg_mse= mse_total/num_images
print(avg_mse)
print(avg_psnr)
print("Average Peak Signal-to-Noise Ratio (PSNR):",avg_psnr, "dB")
```

#NIQE SCORE

```
import cv2
import numpy as np
from scipy import signal
from skimage import feature
import os

def niqe(image):
    # read the image and convert it to grayscale
    img = cv2.imread(image)
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # compute the gradient maps
    sobelx = cv2.Sobel(gray,cv2.CV_64F,1,0,ksize=3)
    sobely = cv2.Sobel(gray,cv2.CV_64F,0,1,ksize=3)
    gradient_map = np.sqrt(sobelx**2 + sobely**2)
```

```
    # compute the mean and variance of the gradient maps
    mu = np.mean(gradient_map)
    sigma = np.std(gradient_map)

   # compute the local statistics
    num_blocks = 10
    block_size = np.floor(np.array(gradient_map.shape)/num_blocks).astype(int)
    mu_map = np.zeros((num_blocks,num_blocks))
    sigma_map = np.zeros((num_blocks,num_blocks))
    for i in range(num_blocks):
        for j in range(num_blocks):
            block =
gradient_map[i*block_size[0]:(i+1)*block_size[0],j*block_size[1]:(j+1)*block_size[1]]
            mu_map[i,j] = np.mean(block)
            sigma_map[i,j] = np.std(block)

    # compute the anisotropy feature
    alpha = 0.03
    rho = 0.5
    a = signal.convolve2d(mu_map**2,np.ones((3,3)),mode='same',boundary='symm')
    b = signal.convolve2d(sigma_map**2,np.ones((3,3)),mode='same',boundary='symm')
    c = signal.convolve2d(mu_map*sigma_map,np.ones((3,3)),mode='same',boundary='symm')
    anisotropy = (a*b - c**2) / (a + b + alpha)
    anisotropy = np.mean(np.maximum(anisotropy,0)**rho)

    # compute the normalized noise power
    noise_map = gray - cv2.GaussianBlur(gray, (7,7), 7/6)
    noise_power = np.mean(noise_map**2) / (np.mean(gradient_map**2) + 1e-5)

    # compute the NIQE score
    c1 = -3.0682
    c2 = 3.8791
    c3 = 1.5946
    c4 = -0.3989
    c5 = 0.00023393
    niqe_score = c1 + c2*np.log(sigma) + c3*anisotropy + c4*np.log(noise_power) +
c5*(np.log(sigma)**2)

    return niqe_score

# set the path to the image dataset
```

```
#image_dir = 'C:/Users/Karmukilan/Pictures/skinnew/clahe images'
image_dir='C:/Users/Karmukilan/Pictures/skinnew/test/nv_median'

# loop over the images in the dataset and compute their NIQE scores
niqe_scores = []
for filename in os.listdir(image_dir):
    if filename.endswith('.jpg'):
        image_path = os.path.join(image_dir, filename)
        score = niqe(image_path)
        niqe_scores.append(score)
        #print('NIQE score for %S')
#print(niqe_scores)
sum(niqe_scores)/len(niqe_scores)
```

#MCACNN

```
import os
import tensorflow as tf
from tensorflow.keras.preprocessing.image import load_img, img_to_array
import numpy as np

# Load all images from a folder
folder_path = "C:/Users/sanch/OneDrive/Desktop/mini project/skinnew/output/"
image_paths = os.listdir(folder_path)
# Create a list to store the loaded images
images = []

# Load and preprocess each image
for image_path in image_paths:
    # Load image
    img = load_img(os.path.join(folder_path, image_path), target_size=(256, 256))
    # Convert image to a NumPy array
    img_array = img_to_array(img)
     # Add the preprocessed image to the list
    images.append(img_array)

# Convert the list of images to a NumPy array
images = np.array(images)

from sklearn.model_selection import train_test_split
train, test = train_test_split(images, test_size=0.5)
model = tf.keras.Sequential([
    #tf.keras.layers.Conv2D(filters, kernel_sizes[i], padding='same', activation='relu')
```

```
    tf.keras.layers.Conv2D(filters=4, kernel_size=(3, 3), activation='relu', input_shape=(256, 256,
3)),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(filters=8, kernel_size=(3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
   tf.keras.layers.Conv2D(filters=16, kernel_size=(3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(filters=32, kernel_size=(3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(filters=64, kernel_size=(3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Conv2D(filters=128, kernel_size=(3, 3), activation='relu'),
    tf.keras.layers.MaxPooling2D((2, 2)),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='softmax')
])

model.compile(optimizer='adam',
            loss='categorical_crossentropy',
            metrics=['accuracy'])
model.fit(train, np.ones(shape=49).astype(int))
predictions = model.predict(test)
```

CHAPTER 4

SNAPSHOTS

Fig. 1.1 Before applying morph blackhat

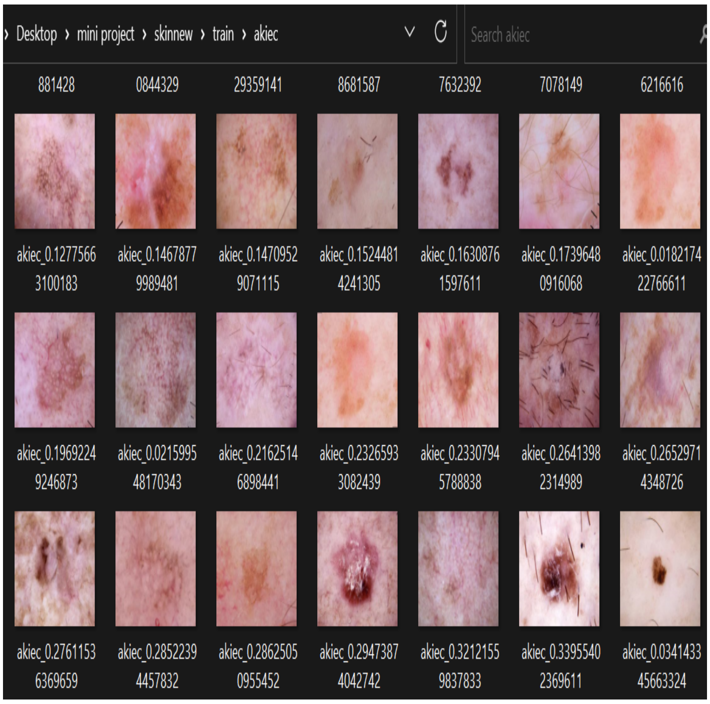


Fig. 1.2 After applying morph black hat

Fig. 1.3 Input for Clahe

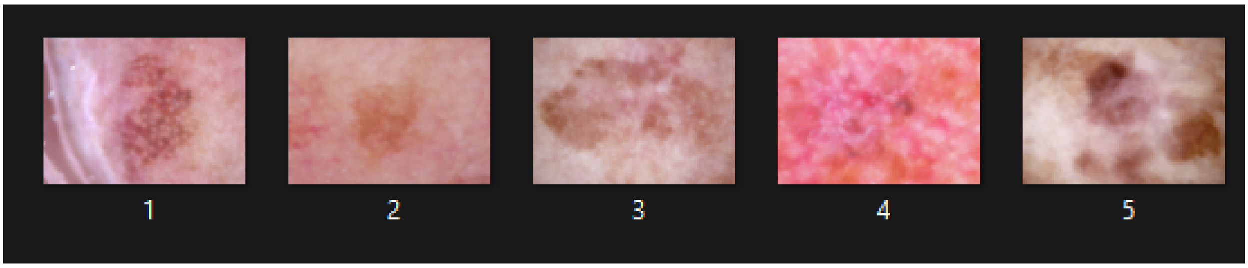


Fig. 1.4 Output of Clahe

Fig. 1.5 Input for bilateral

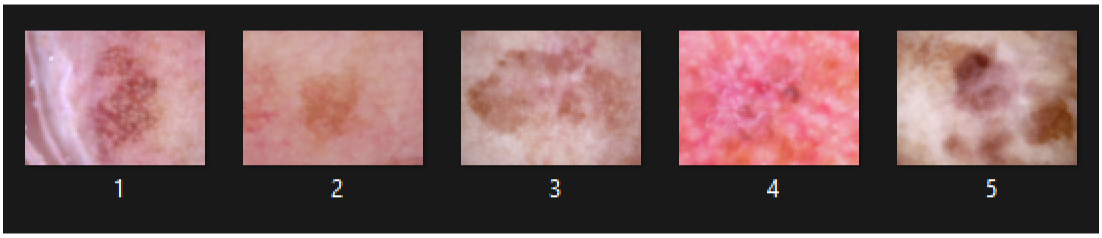


Fig. 1.6 Output of bilateral

Fig. 1.7 Implementation of MCACNN

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 224, 224, 32)      896

 max_pooling2d (MaxPooling2D  (None, 112, 112, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 112, 112, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 56, 56, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 56, 56, 128)       73856

 max_pooling2d_2 (MaxPooling  (None, 28, 28, 128)      0
 2D)

 conv2d_3 (Conv2D)           (None, 28, 28, 128)       147584

 max_pooling2d_3 (MaxPooling  (None, 14, 14, 128)      0
 2D)

 flatten (Flatten)           (None, 25088)             0

 dense (Dense)               (None, 512)               12845568

 dense_1 (Dense)             (None, 7)                 3591

=================================================================
Total params: 13,089,991
Trainable params: 13,089,991
Non-trainable params: 0
_________________________________________________________________
```

```
Epoch 1/10
1332/1332 - 1767s - loss: 1.0870 - accuracy: 0.6503 - val_loss: 0.6345 - val_accuracy: 0.7700 - 1767s/epoch - 1s/step
Epoch 2/10
1332/1332 - 2484s - loss: 0.6006 - accuracy: 0.7875 - val_loss: 0.3671 - val_accuracy: 0.8733 - 2484s/epoch - 2s/step
Epoch 3/10
1332/1332 - 1912s - loss: 0.3768 - accuracy: 0.8667 - val_loss: 0.2703 - val_accuracy: 0.9077 - 1912s/epoch - 1s/step
Epoch 4/10
1332/1332 - 1378s - loss: 0.2314 - accuracy: 0.9211 - val_loss: 0.2217 - val_accuracy: 0.9187 - 1378s/epoch - 1s/step
Epoch 5/10
1332/1332 - 1171s - loss: 0.1345 - accuracy: 0.9556 - val_loss: 0.1232 - val_accuracy: 0.9603 - 1171s/epoch - 879ms/step
Epoch 6/10
1332/1332 - 1081s - loss: 0.0939 - accuracy: 0.9700 - val_loss: 0.1092 - val_accuracy: 0.9657 - 1081s/epoch - 811ms/step
Epoch 7/10
1332/1332 - 983s - loss: 0.0780 - accuracy: 0.9739 - val_loss: 0.0421 - val_accuracy: 0.9857 - 983s/epoch - 738ms/step
Epoch 8/10
1332/1332 - 983s - loss: 0.0573 - accuracy: 0.9815 - val_loss: 0.0416 - val_accuracy: 0.9867 - 983s/epoch - 738ms/step
Epoch 9/10
1332/1332 - 979s - loss: 0.0613 - accuracy: 0.9791 - val_loss: 0.0604 - val_accuracy: 0.9823 - 979s/epoch - 735ms/step
Epoch 10/10
1332/1332 - 1000s - loss: 0.0425 - accuracy: 0.9865 - val_loss: 0.0537 - val_accuracy: 0.9847 - 1000s/epoch - 751ms/step

<keras.callbacks.History at 0x219b1327580>
```

Fig. 1.8 Input for MCACNN

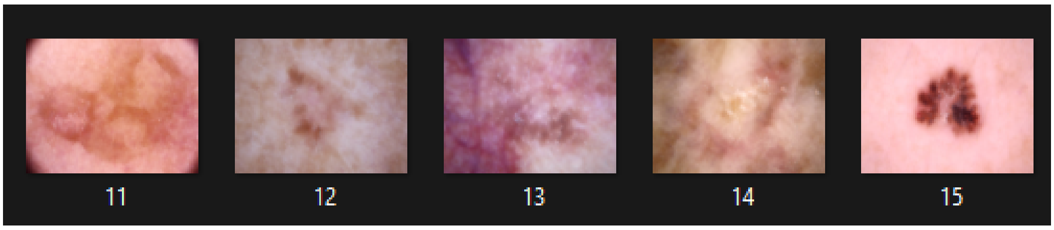
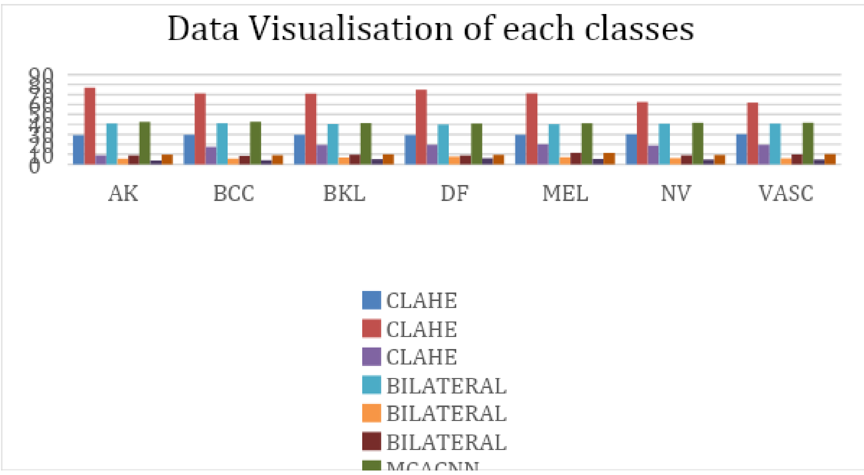
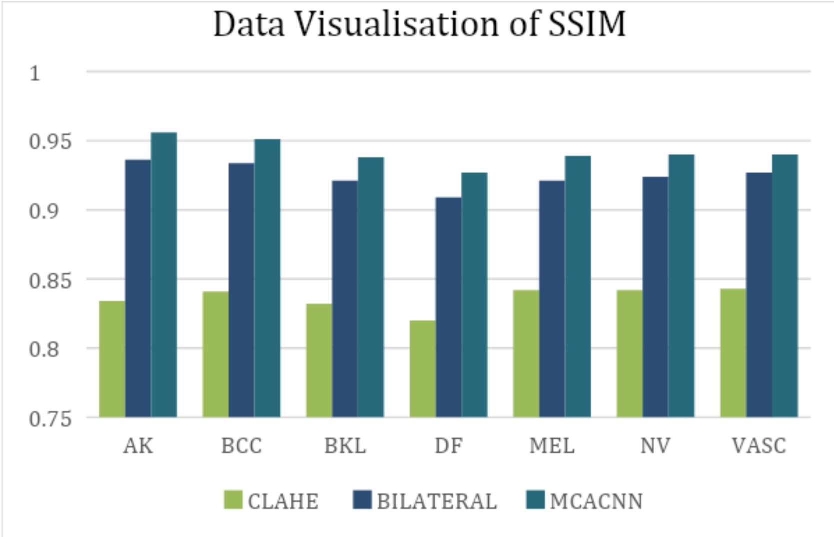
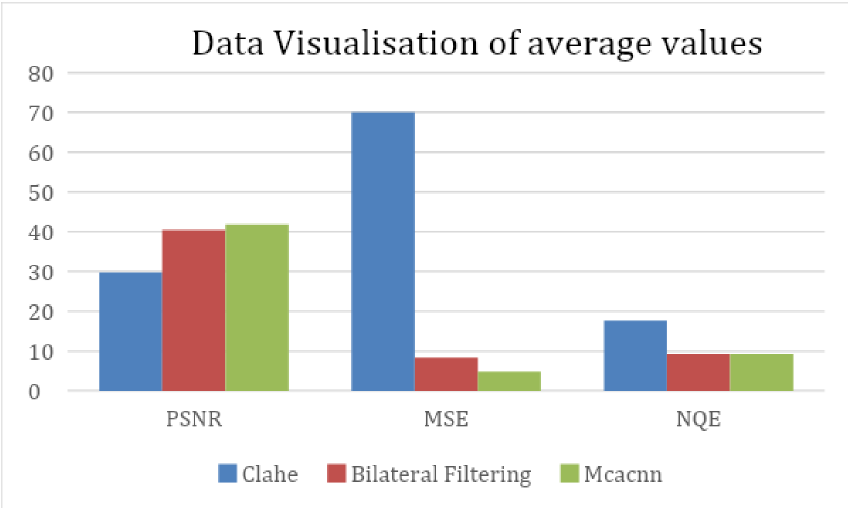


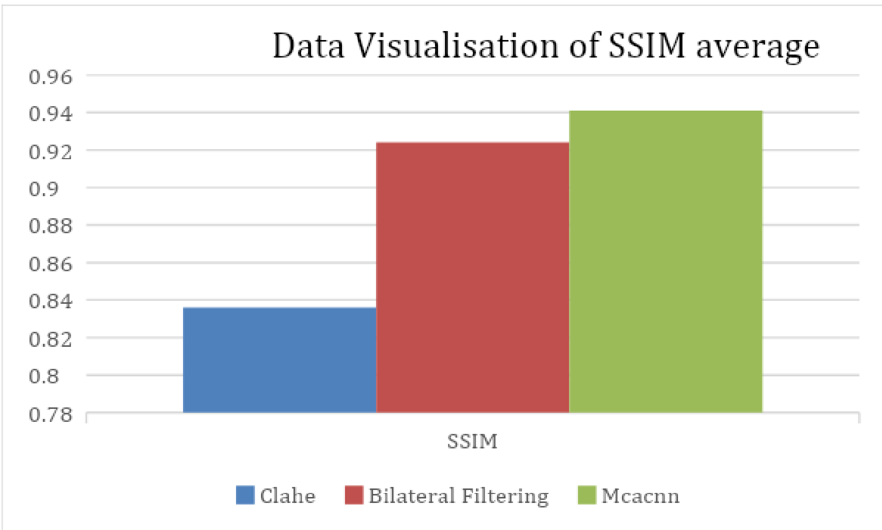
Fig. 1.9 Output of MCACNN

Fig. 1.10 Data visualisation charts




Data Visualisation of SSIM
1
0.95
0.9
0.85
0.8
0.75
AK
BCC
BKL
DF
MEL
NV
VASC
CLAHE
BILATERAL
MCACNN


Data Visualisation of average values
80
70
60
50
40
30
20
10
0
PSNR
MSE
NQE
Clahe
Bilateral Filtering
Mcacnn


Data Visualisation of SSIM average
0.96
0.94
0.92
0.9
0.88
0.86
0.84
0.82
0.8
0.78
SSIM
Clahe
Bilateral Filtering
Mcacnn

CHAPTER 5
CONCLUSION AND FUTURE WORKS

Conclusion:

The project aimed to address hairline obscuration and low contrast in skin images used for melanoma diagnosis. A technique utilizing a CNN with contextual information aggregation was proposed. Evaluated on the HAM10000 dataset, the method outperformed other techniques, enhancing reliability and accuracy. Ablation study confirmed the importance of hairline removal and contextual information. The proposed method shows potential for improving skin melanoma image quality, aiding accurate diagnosis and treatment. Further research and validation on larger datasets are needed, considering privacy and ethical concerns.

Future Works:

1. Including extra modalities like dermoscopy or other imaging methods can add complementary data and boost overall diagnostic precision. Future research might look into combining different modalities to improve the system's performance even more.
2. Patient demographics, lighting conditions, and skin image quality can all vary. Future research can concentrate on creating techniques to manage such heterogeneity and guarantee the model's efficacy across various datasets.
3. CNN models are frequently viewed as "black boxes," which makes it difficult to comprehend how they make decisions. Future studies could look into methods to explain or visualise the model's reasoning, helping dermatologists better comprehend the characteristics and patterns used for melanoma diagnosis.
4. To evaluate the effectiveness of the suggested method in practical settings, extensive validation on larger and more varied clinical datasets is required. Collaboration with healthcare organisations and professionals can make it easier to assess the system's effectiveness and usability in clinical settings.

CHAPTER 6

REFERENCES

1. Wei Li, Alex Noel Joseph Raj, Tardi Tjahjadi, Zhemin Zhuang , Digital hair removal by deep learning for skin lesion segmentation, Volume 117, 2021, 107994, ISSN 0031-3203.
2. Mohamed Attia, Mohammed Hossny, Hailing Zhou, Saeid Nahavandi, Hamed Asadi, Anousha Yazdabadi,Realistic hair simulator for skin lesion images: A novel benchmarking tool,Artificial Intelligence in Medicine, Volume 108, 2020, 101933, ISSN 0933-3657.
3. L. Talavera-Martínez, P. Bibiloni and M. González-Hidalgo, "Hair Segmentation and Removal in Dermoscopic Images Using Deep Learning," in IEEE Access, vol. 9, pp. 2694-2704, 2021,Volume 154,2020,Pages 598-613,ISSN 0960-1481.

CHAPTER 7
APPENDIX

Base Paper Link: **https://doi.org/10.1016/j.bspc.2022.103653**