

**DEEP LEARNING AUGMENTED ASSESSMENT OF SKIN  
PHOTODAMAGE INFORMED BY MULTIPLE DERMATOLOGISTS**

by

**Vladislav Gavril Matibag Marasigan**

**A Thesis**

*Submitted to the Faculty of Purdue University*

*In Partial Fulfillment of the Requirements for the degree of*

**Master of Science in Biomedical Engineering**



Weldon School of Biomedical Engineering

West Lafayette, Indiana

May 2023

**THE PURDUE UNIVERSITY GRADUATE SCHOOL**  
**STATEMENT OF COMMITTEE APPROVAL**

**Dr. Young L. Kim, Chair**

Weldon School of Biomedical Engineering

**Dr. Yunjie Tong**

Weldon School of Biomedical Engineering

**Dr. Michael D. Zoltowski**

Elmore Family School of Electrical and Computer Engineering

**Approved by:**

Dr. Tamara L. Kinzer-Ursem

*Dedicated to my parents*

## **ACKNOWLEDGMENTS**

I would like to express my gratitude to Dr. Young Kim, who took me on as a master's student only last year despite having no research background as well as being a quarter way into my professional master's degree. He was very patient with me in making progress despite all the pivots I've undergone and classes I've had to manage, whether I was the student or the teaching assistant. I would also like to thank the research committee, both Dr. Michael D. Zoltowski and Dr. Yunjie Tong for their willingness to participate in my defense and their guidance along my graduate career. Additionally, I would like to thank my fellow lab members Sang Mok Park, Dr. Jungwoo Leem, Semin Kweon, and Yuhyun Ji for their helpfulness and resourcefulness. Most especially Yuhyun as he handled and guided me in determining metrics and methods in deep learning despite such quick and short turnover times. Finally, I would like to thank my friends in the Weldon School of Biomedical Engineering, Purdue Bells, Salt and Light Christian Fellowship, and Kossuth Street Baptist Church, as well as my parents for their prayers and continued support in my graduate career. I would not have accomplished this much in my graduate career if it weren't for the support and motivation of these people.

## TABLE OF CONTENTS

LIST OF TABLES .....	6
LIST OF FIGURES .....	7
ABBREVIATIONS .....	8
ABSTRACT.....	9
1. INTRODUCTION .....	10
2. MATERIALS AND METHODS .....	13
2.1 Image Collection.....	13
2.2 Dataset Organization and Validation .....	14
2.3 Network Description and Transfer Learning .....	15
2.4 Image Preprocessing and Data Augmentation .....	15
2.5 Network Configurations.....	17
3. RESULTS .....	18
4. CONCLUSION.....	20
APPENDIX.....	21
REFERENCES .....	29

## LIST OF TABLES

Table 1.1. <b>FFPAS developed by McKenzie et al.</b> [3]. Four dimensions are assessed using four categories of severity of skin photodamage particular to patients with actinic keratosis. ....	12
--	----

## LIST OF FIGURES

- Figure 1.1. **Scatterplot of variation for 15 different scores rated by 15 dermatologists from 110 forearm images.** X-axis represents image number and Y-axis represents the standard deviation of the 15 scores rated by the 15 dermatologists. Data was obtained from a clinical study conducted in Wright State University Department of Dermatology. A standard deviation of 1 or more indicates that more than half of the dermatologists do not agree in measurement of skin damage. .... 11
- Figure 2.1. **Examples of forearms with varying degrees of skin photodamage.** Images collected from Wright State University Department of Dermatology. (a) Example of forearm with low severity of skin photodamage. (b) Example of forearm with medium severity of skin photodamage. (c) Example of forearm with high severity of skin photodamage. .... 13
- Figure 2.2. **VGG-16 architecture adapted from Simonyan et al. [7].** Input takes a 224-by-224 sized RGB image. Conv(x)-(y) denotes the convolution filter, with x denoting the x-by-x size of the filter applied to y-number of channels. Maxpool refers to the pooling layer. FC-z is the fully connected layer applied to z-number of channels. Output layer uses a soft-max activation function. .... 15
- Figure 2.3. **Example output of MATLAB square segmentation algorithm.** The algorithm first rotates the original image to straighten out the arm, then crops the relevant portions of the forearm, and finally creates a boundary mask of the arm. Square segments are generated and spaced in a way that maximizes the amount of area covered in the forearm while maintaining consistent sizes for each forearm sample. This example shows the right hand of the 11<sup>th</sup> subject. The expected file name for the purple square is “11R\_3”. .... 16
- Figure 3.1 **Confusion matrix comparison of models.** Blue areas denote correct identification of labels. Red areas denote misidentification of the labels. (a) Confusion matrix for neural network model predictions. (b) Confusion matrix of dermatologist ratings. .... 19

## **ABBREVIATIONS**

NMSC – non-melanoma skin cancer

UV – ultraviolet

BCC – basal cell carcinoma

SCC – squamous cell carcinoma

CNN – convolutional neural network

FFPAS – Dermatologic Assessment Form Forearm Photographic Assessment Scale

TP – true positive

TN – true negative

FP – false positive

FN – false negative



## **ABSTRACT**

Non-melanoma skin cancers are primarily caused by ultraviolet radiation and affects a large population of the United States. The only available tool to assess skin photodamage is the McKenzie scale. However, the subjective and qualitative nature of this method leads to variability and inconsistency among dermatologists. We propose applying a deep learning approach to address this issue. 55 patients were assessed by 15 board-certified dermatologists rating the degree of skin photodamage using the McKenzie scale. Using a pretrained convolutional neural network, we train and test a model on labeled forearm images classified based on the severity of photodamage. We employ image preprocessing and data augmentation to the dataset as well as configure parameters and hyperparameters of the network architecture to obtain the optimal model to predict the degree of photodamage on the skin. Cross validation is performed to ensure the practical effectiveness of the model. Finally, performance of the neural network model is compared to that of the dermatologist ratings to determine feasible application of this model. We envision this as augmented technology for objective and reliable assessment of skin photodamage for dermatologists.

## 1. INTRODUCTION

It is estimated that non-melanoma skin cancer (NMSC) affects close to 5.4 million persons in the United States [1]. Ultraviolet (UV) radiation has the largest impact as a risk factor for skin cancer, linked to two of the most common types of NMSC: basal cell carcinoma (BCC) and squamous cell carcinoma (SCC) [2]. BCC and SCC occur on cosmetically sensitive areas such as the face and ears, leading to additional affliction. The total cost of treating NMSC is \$650 million in the United States, yielding a heavy burden on the healthcare system in the United States. As such, early detection and prevention of NMSC is vital in reducing costs and morbidity by accurate and reliable assessment of skin photodamage skin damage.

To accurately diagnose skin photodamage to effectively treat patients, there is a need for a consistent and reliable standard of measurement. This need has led to the development of an assessment scale to measure the degree of photodamage of the skin using descriptive, visual, and photographic grading scales [3]. However, relying on traditional methods such as visual inspection or subjective grading scales can be time-consuming, costly, and invasive should they require skin biopsies. These methods are also qualitative in nature and do not have clear quantitative markings for measurement. As a result, there exists a substantial amount of disagreement and variability amongst clinicians as shown in Figure 1.1 [4].

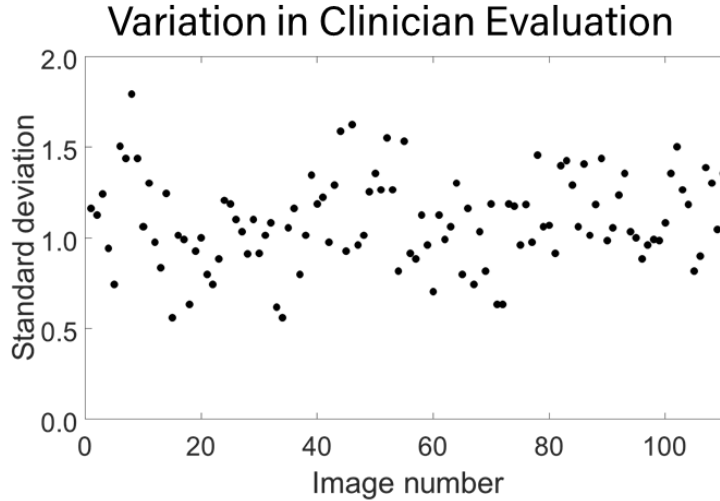


Figure 1.1. **Scatterplot of variation for 15 different scores rated by 15 dermatologists from 110 forearm images.** X-axis represents image number and Y-axis represents the standard deviation of the 15 scores rated by the 15 dermatologists. Data was obtained from a clinical study conducted in Wright State University Department of Dermatology. A standard deviation of 1 or more indicates that more than half of the dermatologists do not agree in measurement of skin damage.

The advent of machine learning methods, specifically deep learning, marks potential for quantitative ways for effective diagnoses in dermatology [5]. Convolutional neural networks (CNN) have great potential in object detection and image classification. Studies have found that CNNs have good performance in classifying dermatological diseases [6]. These methods can not only improve accuracy but save time and reduce the burden of work for dermatologists.

This study investigates the performance of using deep learning methods to classify UV skin photodamage to improve accuracy and effectiveness of assessment for better diagnosis and treatment of skin diseases. 55 bilateral arm samples are used to assess the severity of photodamage totaling 110 arm images. 15 board-certified dermatologists evaluated the arm samples using the global assessment of the Dermatologic Assessment Form Forearm Photographic Assessment Scale (FFPAS) across four criteria as shown in Table 1.1. Deep learning methods in this study are used to obtain a more objective and reliable score for severity of skin photodamage by minimizing the variability of the dermatologist scores.

**Table 1.1. FFPAS developed by McKenzie et al. [3].** Four dimensions are assessed using four categories of severity of skin photodamage particular to patients with actinic keratosis.

Clinical Sign	Absent	Mild	Moderate	Severe
Fine wrinkling	0	1 2 3	4 5 6	7 8 9
Coarse wrinkling	0	1 2 3	4 5 6	7 8 9
Abnormal Pigmentation	0	1 2 3	4 5 6	7 8 9
Global	0	1 2 3	4 5 6	7 8 9

## 2. MATERIALS AND METHODS

### 2.1 Image Collection

In accordance with an institutional review board-approved protocol, a clinical study was conducted on patients from clinics from Wright State University Department of Dermatology. These patients were at least 35 years old, had fair skin (Fitzpatrick scale I or II), and had not used a tanning bed or had significant exposure to the sun within six months of measurement. Examples are shown in Figure 2.1.

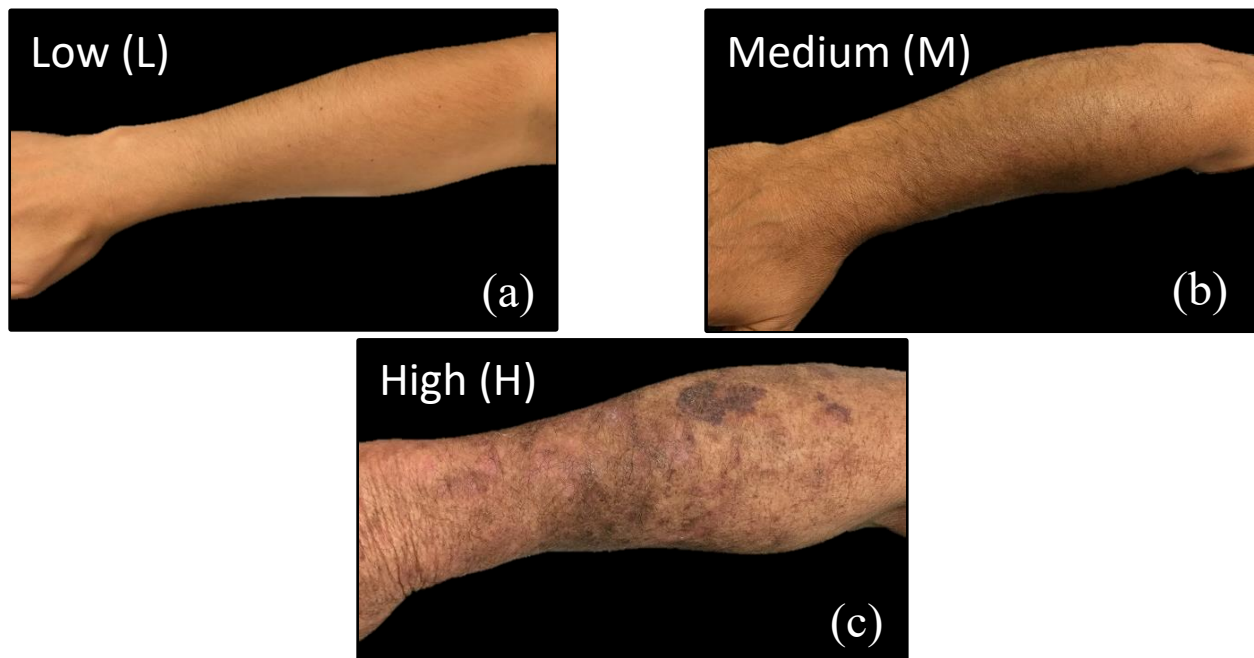


Figure 2.1. **Examples of forearms with varying degrees of skin photodamage.** Images collected from Wright State University Department of Dermatology. (a) Example of forearm with low severity of skin photodamage. (b) Example of forearm with medium severity of skin photodamage. (c) Example of forearm with high severity of skin photodamage.

The study recruited 55 subjects with varying degrees of skin photodamage. Bilateral photographs of their forearms were taken for a total of 110 images. Clinical signs of UV skin photodamage were assessed using the 10-point FFPAS by McKenzie that scores based on four categories: fine wrinkling, coarse wrinkling, abnormal pigmentation, and a global assessment. 15 board-certified dermatologists, including 4 from academic and 11 from private practice backgrounds who had a minimum of 5 years of post-residency experience, independently

evaluated each arm of the participants using this scale. They were trained by examples provided by the McKenzie scale. Afterwards, the dermatologists individually, independently, and separately evaluated each arm of the subject from a PowerPoint presentation of 110 forearms of the 55 subjects with an additional 20 forearm pictures duplicated to determine intra-rater reliability. Dermatologists were given an unlimited amount of time to assess the samples with knowledge of patient nor arm identification, clinical information, sources of photos. The dermatologists were also not allowed to discuss their observations with each other. A total of 1,640 scores (110 images  $\times$  15 dermatologists) were obtained across all samples.

## **2.2 Dataset Organization and Validation**

An ensemble approach was utilized to categorize and organize dermatologist scores for each bilateral forearm sample to represent the 15 observations of UV skin damage. First, the mode of each sample amongst the dermatologist ratings was taken to represent the most frequent score of the arms. This was done as scores of the dermatologists were inconsistent, and a reasonable approach to determine the ground truth of the images was needed. Then, UV skin photodamage was discretized into 3 categories: low, medium, and high. This was based on the McKenzie scale classification: 0 is no actinic damage, 1-3 are low, 4-6 are moderate/medium, and 7-9 being severe/high. Because there were very few data samples for no actinic damage (0), forearms classified as such were reclassified as low.

Cross-validation is performed to determine how accurately the model will perform in practice and prevent overfitting. This is done by splitting the data into training sets to be used in training the model and testing sets to compare the model predictions to the actual data. The k-fold approach is used, dividing the data into 11 subsets and performing the training and validation 11 times. The data was randomly partitioned into a 10:1 ratio of training and testing data. This was done in such a way that each score classification group were partitioned proportionally to their population and that at each data sample was included exactly once in a testing set among all the folds. The 11 results for each fold are then averaged to determine the performance of the model.

### 2.3 Network Description and Transfer Learning

VGG-16 is a convolutional neural network used for object detection and image classification. It is configured to a depth of 16 layers and has an image input size of 224-by-224 pixels [7]. Due to a smaller working dataset, the transfer learning technique will be used. This involves using networks already trained on a previous dataset, then applying the knowledge gained to perform a different task [8]. This will improve the model by relying on a larger, pre-existing dataset and applying the gained knowledge to a smaller dataset. Our pretrained model of VGG-16 comes from MathWorks Deep Learning library. It is trained on the ImageNet database, containing over 14 million labeled images [9].

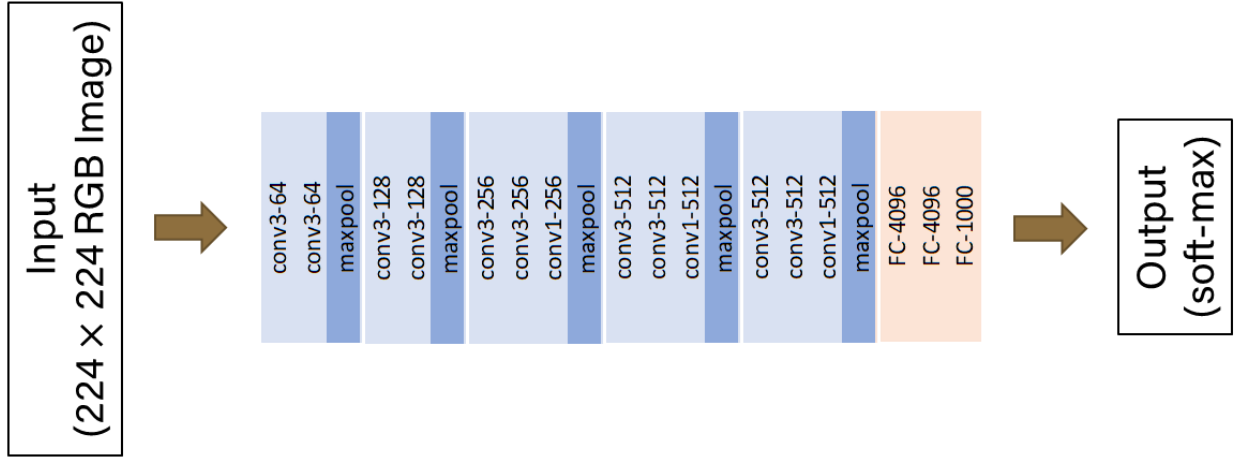


Figure 2.2. **VGG-16 architecture adapted from Simonyan et al.** [7]. Input takes a 224-by-224 sized RGB image. Conv(x)-(y) denotes the convolution filter, with x denoting the x-by-x size of the filter applied to y-number of channels. Maxpool refers to the pooling layer. FC-z is the fully connected layer applied to z-number of channels. Output layer uses a soft-max activation function.

### 2.4 Image Preprocessing and Data Augmentation

To train the pretrained network on the new dataset, images must be labeled and preprocessed to the model's requirements. Images in the dataset were taken with different resolutions under the JPEG format but because VGG-16 only takes 224-by-224 size inputs, the images must be scaled. Rather than scale every sample, 4 square segments of each forearm in the image (shown in Figure 2.3) were taken to increase sample size and decrease noise from outside objects in an image. This was done by developing an automated segmentation algorithm in MATLAB. Afterwards, all square segments are resized to 224-by-224 to match the input size. The preprocessed images are

then stored in a folder corresponding to their categorization (High, Medium, Low). This is done for 11 iterations corresponding to each fold in the k-fold cross validation process, sifting them based on training and testing data. As such, 11 folders for testing data and training data each containing folders for each of their categorization are produced for a total of 22 folders. The naming convention of each image is as follow: “<subject number><hand>\_<segment number>”.

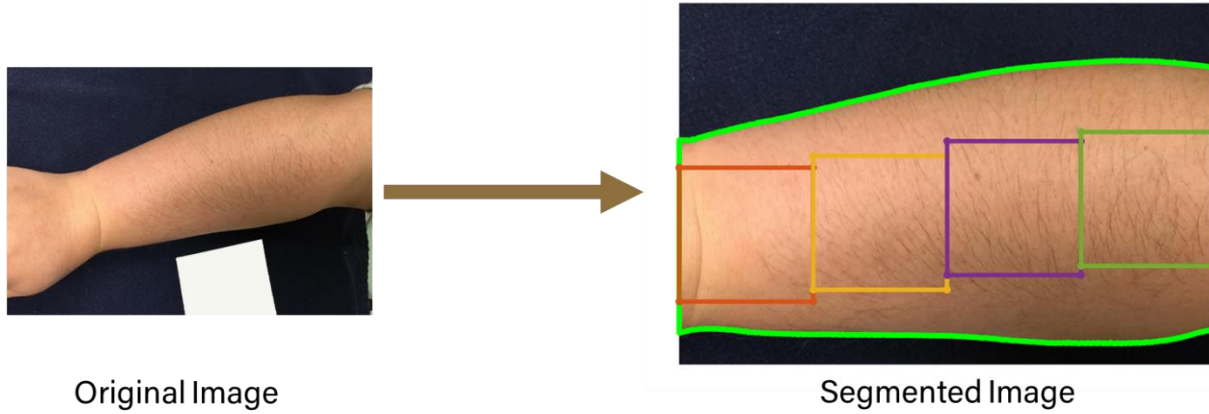


Figure 2.3. **Example output of MATLAB square segmentation algorithm.** The algorithm first rotates the original image to straighten out the arm, then crops the relevant portions of the forearm, and finally creates a boundary mask of the arm. Square segments are generated and spaced in a way that maximizes the amount of area covered in the forearm while maintaining consistent sizes for each forearm sample. This example shows the right hand of the 11<sup>th</sup> subject. The expected file name for the purple square is “11R\_3”.

To improve performance, data augmentation techniques were used. This was done with the following techniques:

1. Random Scale: Randomly scales the input image increasing or decreasing by 50%.
2. Random X-Reflection: Randomly flips the input image horizontally with a probability of 0.5.
3. Random Y-Reflection: Randomly flips the input image vertically with a probability of 0.5.
4. Random Rotation: Randomly rotates the input image by an angle between -45 to 45 degrees.

To ensure dataset consistency in model training, each set of segments are grouped together in each fold in cross validation when partitioning the images into testing and training data.



## 2.5 Network Configurations

Configurations of the neural network were adjusted to improve performance and accuracy. Mini batch size was set to 32 to improve generalization performance [10]. After testing and iterating the model multiple times, a learn rate of 0.0001 was found to be most optimal. A maximum of 64 epochs was chosen to ensure the neural network converges after enough iterations. The algorithm used to optimize training is a stochastic gradient descent with momentum. The model was trained in MATLAB using the MATLAB Deep Learning Library running on an Intel Core i9-10900X 3.7GHz CPU, single Nvidia GeForce RTX 3090 GPU, and 32 GB RAM. Once the neural network has run, the model will classify images from the testing set and accuracy is determined by calculating the percentage of correct identification by label per image. Because 4 images are produced for each subset of a sample, based on the dominating label classified in the set of 4. If at least one of the images in the set is classified as medium in a set containing low labels as well, the set is classified as medium. Likewise, if at least one square is labeled as high, the whole entire set is labeled high. This is done because skin photodamage throughout a forearm is not necessarily homogenous throughout an entire forearm. Certain parts of a forearm may exhibit higher degrees of photodamage than other parts; however, dermatologists would label such a sample based on the highest severity.

### 3. RESULTS

A 3-by-3 confusion matrix is used to evaluate the performance of the model. The matrix details the performance of the model by comparing the predicted and actual labels from the testing set of data. Generally, there are four elements to the confusion matrix: true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN). True positives and negatives correspond to the correct classification of an existing or lack of a label. False positives and negatives correspond to the incorrect classification of an existing or lack of a label. The 3-by-3 matrix contains 9 categories: 3 for each correct classification of each label and 6 for each permutation of incorrect classification. Training and testing results from the 11 folds of cross validation are used to create the confusion matrix for the neural network model while scores from the 15 dermatologists were used for the dermatologist ratings. To quantify the performance of the neural network model, the precision and recall of the model is calculated and compared to that of the dermatologist ratings. Precision quantifies the number of correct predictions of a specific class that belong to that specific class, calculated as,

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

Recall quantifies the number of correct predictions of a specific class from all samples of that specific class in a dataset, calculated as,

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

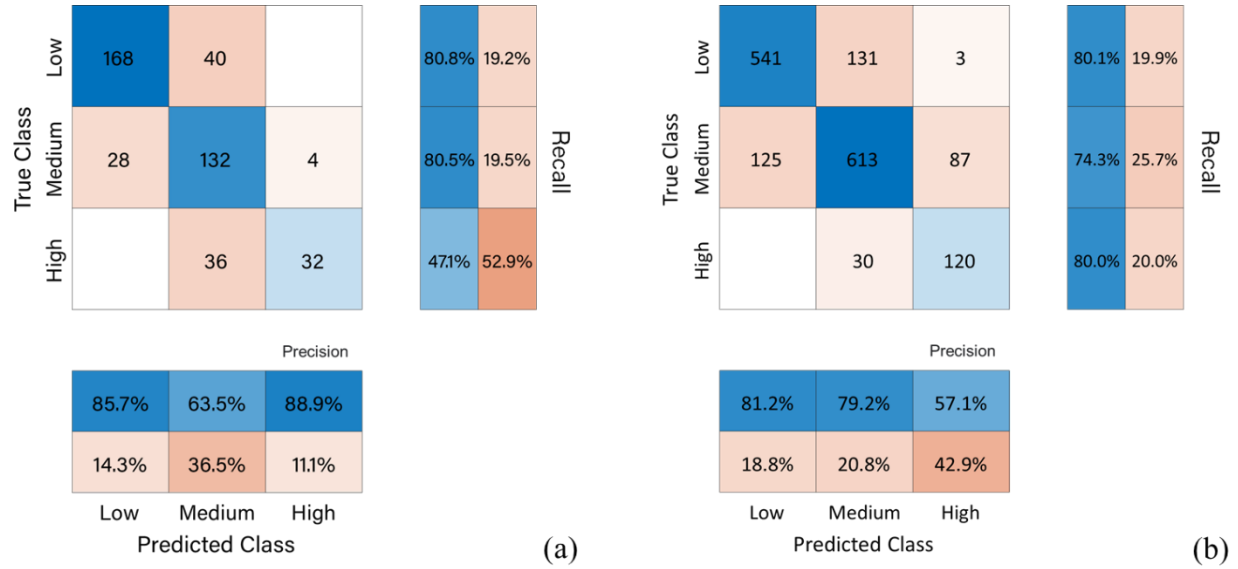


Figure 3.1 **Confusion matrix comparison of models.** Blue areas denote correct identification of labels. Red areas denote misidentification of the labels. (a) Confusion matrix for neural network model predictions. (b) Confusion matrix of dermatologist ratings.

As shown in Figure 3.1, the neural network model improves on the dermatologist ratings on precision of identifying low (85.7% vs. 81.2%) and high (88.9% vs. 57.1%) severities as well as recall of low (80.8% vs. 80.1%) and medium (80.5% vs. 74.3%) severities; however, the neural network model fails to improve in precision on identifying medium severities (63.5% vs. 79.2%) and recall of high severities (47.1% vs. 80.0%).

## **4. CONCLUSION**

This study used a total of 110 image samples of bilateral forearms from 55 patients to determine the extent of photodamage from UV radiation. 15 dermatologists evaluated the images using the Global Assessment Severity Scale to assess the clinical signs of UV photodamage. The dermatologist ratings were grouped into low, medium, and high severities of photodamage, and a model-based approach was used to estimate the accurate score for each bilateral forearm. Cross validation through training and testing was done to determine the practical performance of the model. The goal of the study was to reduce subjectivity and variability amongst dermatologist assessment by using deep learning methods through MathWorks' pretrained VGG-16 neural network for image classification to obtain a less biased and objective score of the severity of photodamage.

## APPENDIX

The following code and functions are used to perform preprocessing methods and executing the deep learning neural network.

### Segmentation\_v3 – segments forearm images into 4 square segments

```
clear
disp('select images folder')
selpath = uigetdir;
folder = dir(selpath);
disp('select save folder')
save_path = uigetdir;

f = input('Choose File number to start (3 or above, 3 starts at the beginning):');
for file_num = f:1:112
    disp(file_num)
    image = imread(folder(file_num).name);
    export_gen = erase(folder(file_num).name, '.JPG'); % deletes .jpg
    export_gen = [export_gen '_'];
    file_type = '.JPG';

    % delete margins
    black_row = find(image(:,1,1)<2 & image(:,1,2)<2 & image(:,1,3)<2); % get the
block bars margins
    image(black_row, :, :) = [];
    image(1:15, :, :) = []; % in case bar is glitchy
    image(end-15:end, :, :) = []; % in case last bar is glitchy

    imshow(image)
    axis off

    rotate_image = inc_rotate(image); % function to incrementally rotate image

    imc = imcrop(rotate_image);
    imc_R = imc(:, :, 1); % Red values have higher contrast
    imc_R_i = imcomplement(imc_R); % inversion

    BW = inc_tol(imc_R_i); % function to incrementally increase or decrease tolerance
for masking

    imshow(BW)

    dim = size(BW);
    col = round(dim(2)/2)-90;
    row = min(find(BW(:, col))));

    boundary = bwtraceboundary(BW, [row, col], 'N');
    N = 4; % square number
```

```

squares = create_squares(boundary,N); % (x,y)

imshow(imc)
hold on;
scatter(boundary(:,2),boundary(:,1),'g','LineWidth',4);
axis on

% square creation
DL_size = 224; % vgg16 size, change according to NN
for x = 1:N
    selection = squares(x).square;
    x_range = min(selection(:,1)):max(selection(:,1));
    y_range = min(selection(:,2)):max(selection(:,2));
    im_sq = imc(y_range,x_range,:); % still flipped in image
    im_sq = imresize(im_sq,[DL_size DL_size]); % resize for vgg16
    plot(selection(:,1),selection(:,2), '-o','LineWidth',8);
    axis off
    % need to export images
    im_export_name = [save_path '\\' export_gen sprintf('%i',x) file_type];
    imwrite(im_sq,im_export_name)
end
pause
close
end

```

**inc\_rotate** – function that incrementally rotates image by user input with goal to “straighten” forearm to maximize square area

```

function rt_image = inc_rotate(image)
% incrementally rotates image based on user input

orientation = -1;
rotate_image = imrotate(image, orientation);

rotation = 0; % track number of rotations
sw = 1;
rotation(1,sw) = rotation(1,sw) + orientation;

cont = [];
while isempty(cont)
    rotate_image = imrotate(rotate_image,orientation);
    rotation(1,sw) = rotation(1,sw) + orientation;
    hold off
    imshow(rotate_image)
    hold on
    yline(size(rotate_image,1)/2,'c')
    axis on

    cont = input('2: switch, 1: stop, else enter to cont: ');
    if cont == 2
        orientation = -orientation;
        cont = [];
        sw = sw + 1;
    end
end

```

```

        rotation(1,sw) = 0;
    end
end
hold off
rt_image = imrotate(image, sum(rotation));
disp('....')
end

```

**inc\_tol** – *function that incrementally changes tolerance image by user input to obtain ideal forearm mask*

```

function BW_image = inc_tol(image)

tol = 0.5;
inc = 0.01; % increasing
cont = [];

while isempty(cont)
    BW_image = im2bw(image,tol);
    BW_image = imcomplement(BW_image);
    imshow(BW_image)
    axis on
    cont = input('2: switch tol direction, 1: stop, 8: invert, enter to cont:');
    if cont == 2
        inc = -inc;
        cont = [];
    elseif cont == 8
        image = imcomplement(image);
        cont = [];
    elseif cont == 1
        cont = 1;
    end
    tol = tol + inc;
end
end

```

**create\_squares** – *function that creates appropriately sized squares to maximize relevant area of forearm*

```

function sq = create_squares(boundary, N)
% boundary: based on produced boundary from bwtraceboundary
% N: number of squares (works best with 4 squares)

length_arm = max(boundary(:,2)) - min(boundary(:,2)); % expected x

out_of_bounds = 1;
iteration = 0;
while out_of_bounds
    out_of_bounds = 0; % reset default
    close

```

```

% create square
sq_len = length_arm/N - iteration; % side of square, decreasing in iteration,
(x,y)
x_b_mid = (length_arm/N - sq_len)/2; % x boundary mid points
% corners
ul_corner = [1 sq_len];
ur_corner = [sq_len sq_len];
ll_corner = [1 1];
lr_corner = [sq_len 1];
sq_plot = [ul_corner; ur_corner; lr_corner; ll_corner; ul_corner]; % last
ul_corner to connect

sq_y_mid = floor(sq_len/2); % y midpoint of square
sq_x_mid = sq_y_mid; % x midpoint of square

new_b = boundary; % (y,x)

% delete sides
for a = 1:2 % 2 bounds
    new_b_x = new_b(:,2);
    new_b_y = new_b(:,1);
    dup_x = new_b_x == mode(new_b_x);
    new_b(dup_x,:) = []; % delete sides here
end

% separate top and bottom bounds using kmeans
idx = kmeans(new_b(:,1),2);
k1 = [new_b((idx==1),2) new_b((idx==1),1)];
k2 = [new_b((idx==2),2) new_b((idx==2),1)];

% check which becomes top or bot (note switch to x,y)
if mean(k1) > mean(k2)
    top_b = k1;
    bot_b = k2;
else
    bot_b = k1;
    top_b = k2;
end

it_ch = 0; % initialize change in iteration
for s = 1:N
    if s == 1
        add_len = x_b_mid + sq_len/2;
    else
        add_len = add_len + 2*x_b_mid + sq_len;
    end
    half_len = add_len - sq_len/2;
    % segment area into slices based on number of squares
    slice_top = top_b(:,1) >= (s + half_len) & top_b(:,1) < (sq_len + half_len);
    slice_bot = bot_b(:,1) >= (s + half_len) & bot_b(:,1) < (sq_len + half_len);
    xline(s + half_len)
    xline(sq_len + half_len)

    top_b_min = min(top_b(slice_top,2));
    bot_b_max = max(bot_b(slice_bot,2));

```



```

top_bot_range = top_b_min - bot_b_max;

y_b_mid = floor((top_bot_range)/2 + bot_b_max); % mid point of y boundary
area
sq_y = floor(sq_plot(:,2) + (y_b_mid - sq_y_mid)); % adjusted y values
sq_x = floor(sq_plot(:,1) + add_len - sq_len/2);

new_sq = [sq_x sq_y]; % (x,y)

if top_bot_range < sq_len % if any instance of square is bigger than slice
    out_of_bounds = 1;
    if top_bot_range > it_ch
        it_ch = sq_len - top_bot_range;
    end
end
sq(s).square = new_sq;
end
iteration = iteration + it_ch; % adjust length of square that maximizes area of
the square while maintaining consistent square size
end

```

**move\_files** – moves segmented images and partitions them into categorized folders based on label

```

disp('select images folder')
selpath = uigetdir;
folder = dir(selpath);
disp('select save folder')
save_path_1 = uigetdir;
save_path_2 = uigetdir;
save_path_3 = uigetdir;

T = readtable('GT and Cross validation copy.xlsx','Range','A2:C112');
hand = T.Hand;
hand = string(hand);
sub = T.Subject;
res = T.ModeGT;

p = length(hand);
i = 1;

for x = 3:442
    fn = folder(x).name;
    ff = folder(x).folder;
    export_name = [ff '\ ' fn];
    hand_ind = find(fn == 'L' | fn == 'R');
    hand_mode = fn(hand_ind);
    sub_num = str2num(fn(1:hand_ind-1));
    T_ind = find(hand == hand_mode & sub == sub_num);
    mode = res(T_ind);

    if mode == 1
        movefile(export_name,save_path_1)
    elseif mode == 2
        movefile(export_name,save_path_2)
    end
end

```

```

elseif mode == 3
    movefile(export_name,save_path_3)
end
i = i + 1;
if i == 4
    i = 1;
end
end
end

```

**rename\_files** – *Renames segments to appropriate naming convention*

```

disp('select images folder')
selpath = uigetdir;
folder = dir(selpath);
disp('select save folder')
save_path = uigetdir;

for x = 3:442
    fn = folder(x).name;
    image = imread(fn);
    fn(end-3:end) = []; % delete .JPG
    new_name = fn;
    if fn(1) == 'A'
        new_name = erase(fn,'A.I. '); % delete A.I.
        hand_ind = find(new_name=='a'|new_name=='b');
        new_name(3:hand_ind-1) = [];
    else
        % delete characters before sub number
        space = max(find(new_name==' '));
        new_name(1:space) = [];
    end
    if new_name(end-2) == 'b'
        new_name(end-2) = 'R';
    elseif new_name(end-2) == 'a'
        new_name(end-2) = 'L';
    end
    new_name = [save_path '\' new_name '.JPG'];
    imwrite(image,new_name)
end

```

**pdnn** – *neural network script to initiate training and exports prediction and testing data.*

```

test_fold = categorical([]);
pred_fold_1 = categorical([]);
%% separate dataset to testing and training (make sure kfolder is open)
for x = 1:11
    i = 1; % iteration
    train_name = sprintf('training_%i',x);
    test_name = sprintf('testing_%i',x);

    inputSize = [224 224];

```

```

trainingImages = imageDatastore(train_name, ...
    'IncludeSubfolders',true, ...
    'LabelSource','foldernames');

% augment data
augmenter = imageDataAugmenter(...
    'RandScale',[0.5 2], ...
    'RandXReflection', true, ...
    'RandYReflection', true, ...
    'RandRotation',[-45 45]);

trainingImagesAug = augmentedImageDatastore([inputSize
3],trainingImages,'DataAugmentation',augmenter);

% test images
testImages = imageDatastore(test_name, ...
    'IncludeSubfolders',true, ...
    'LabelSource','foldernames');

% trainingImages.ReadFcn = @(loc)imresize(imread(loc),inputSize);
testImages.ReadFcn = @(loc)imresize(imread(loc),inputSize);

%% pretrained neural network
net = vgg16;
layers = net.Layers;

%% Modify to use 3 categories
layers(39) = fullyConnectedLayer(3);
layers(41) = classificationLayer;

%% retrain network
opts = trainingOptions('sgdm','InitialLearnRate',
0.0001,'MaxEpochs',32,'MiniBatchSize',64);
myNet = trainNetwork(trainingImagesAug,layers, opts);

%% Output for classification
% first method -> normal accuracy check
predictedLabels = classify(myNet, testImages);

test_fold(:,x) = testImages.Labels;
pred_fold_1(:,x) = predictedLabels;
end
filename = sprintf('aug_mat_%i.mat',i);
save(filename,'pred_fold_1','test_fold')
i = i + 1;

```

**accuracy\_check\_v3** – to be executed after running *pdnn*; calculates the accuracy and produces confusion matrix based on results of neural network

```

%% accuracy checking
for x = 1:11
    set = categorical([]);
    method = categorical([]);

```

```

L = length(pred_fold_1);
for y = 0:(L/4)-1
    set = pred_fold_1((1+4*y):(4+4*y),x);
    if any(set == 'High')
        method((1+4*y):(4+4*y)) = 'High';
    elseif any(set == 'Medium')
        method((1+4*y):(4+4*y)) = 'Medium';
    else
        method((1+4*y):(4+4*y)) = 'Low';
    end
end
accuracy = mean(method' == test_fold(:,x));

pred_fold_2(:,x) = method';

acc_fold(x) = accuracy;
end

avg_acc = mean(acc_fold);

test_c = test_fold(:);
pred_c2 = pred_fold_2(:);

[c2,o2] = confusionmat(pred_c2,test_c, 'order',{'Low','Medium','High'});

figure('Renderer', 'painters', 'Position', [10 10 900 900]);
cm2 = confusionchart(c2, o2);
cm2.NormalizedValues;
cm2.RowSummary = 'row-normalized';
cm2.ColumnSummary = 'column-normalized';

```

## REFERENCES

- [1] H. W. Rogers, M. A. Weinstock, S. R. Feldman, and B. M. Coldiron, “Incidence Estimate of Nonmelanoma Skin Cancer (Keratinocyte Carcinomas) in the US Population, 2012,” *JAMA Dermatol*, vol. 151, no. 10, pp. 1081–1086, Oct. 2015, doi: 10.1001/JAMADERMATOL.2015.1187.
- [2] J. D’Orazio, S. Jarrett, A. Amaro-Ortiz, and T. Scott, “UV Radiation and the Skin,” *Int J Mol Sci*, vol. 14, no. 6, p. 12222, 2013, doi: 10.3390/IJMS140612222.
- [3] N. E. McKenzie, K. Saboda, L. D. Duckett, R. Goldman, C. Hu, and C. N. Curiel-Lewandrowski, “Development of a photographic scale for consistency and guidance in dermatologic assessment of forearm sun damage,” *Arch Dermatol*, vol. 147, no. 1, pp. 31–36, Jan. 2011, doi: 10.1001/ARCHDERMATOL.2010.392.
- [4] B. Schmeusser *et al.*, “Inter- and Intra-physician variation in quantifying actinic keratosis skin photodamage,” *J Clin Investig Dermatol*, vol. 8, no. 2, pp. 1–4, Dec. 2020, doi: 10.13188/2373-1044.1000065.
- [5] S. Chan, V. Reddy, B. Myers, Q. Thibodeaux, N. Brownstone, and W. Liao, “Machine Learning in Dermatology: Current Applications, Opportunities, and Limitations,” *Dermatol Ther (Heidelb)*, vol. 10, no. 3, pp. 365–386, Jun. 2020, doi: 10.1007/S13555-020-00372-0/FIGURES/3.
- [6] A. K. Sah, S. Bhusal, S. Amatya, M. Mainali, and S. Shakya, “Dermatological Diseases Classification using Image Processing and Deep Neural Network,” *Proceedings - 2019 International Conference on Computing, Communication, and Intelligent Systems, ICCIS 2019*, vol. 2019-January, pp. 381–386, Oct. 2019, doi: 10.1109/ICCIS48478.2019.8974487.
- [7] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, Sep. 2014, Accessed: Apr. 10, 2023. [Online]. Available: <https://arxiv.org/abs/1409.1556v6>
- [8] F. Zhuang *et al.*, “A Comprehensive Survey on Transfer Learning,” *Proceedings of the IEEE*, vol. 109, no. 1, pp. 43–76, Jan. 2021, doi: 10.1109/JPROC.2020.3004555.
- [9] “ImageNet.” <https://image-net.org/> (accessed Apr. 16, 2023).

- [10] D. Masters and C. Lusch, “Revisiting Small Batch Training for Deep Neural Networks,” Apr. 2018, Accessed: Apr. 16, 2023. [Online]. Available: <https://arxiv.org/abs/1804.07612v1>