A Brief Inquiry Into Supervised Learning In Skin Cancer: Asymmetry, ROI, And KNN Classification

Adler Antonio Calvillo Arellano
AI Engineering Student
Universidad Panamericana
Aguascalientes, México
anthony_adler@outlook.com

Jared López García
AI Engineering Student
Universidad Panamericana
Aguascalientes, México
jared4lg@gmail.com

Max Georges Sainte Guzmán
AI Engineering Student
Universidad Panamericana
Aguascalientes, México
max77sab@gmail.com

***Abstract*—This study presents the development of a machine learning model for the classification of skin lesions using the HAM10000 dataset. The primary objective is to enhance early detection and diagnosis of skin cancer through the application of computer vision and supervised learning techniques. We explore various methodologies including the extraction of RGB features from dermatoscopic images, implementation of Principal Component Analysis (PCA) for dimensionality reduction, and the use of K-Nearest Neighbors (KNN) for classification. Additionally, we compare the performance of models using raw images against those focusing on the region of interest (ROI) to improve diagnostic accuracy and analyzed ground-truth data in relation to lesion asymmetry. Also the incorporation of the One vs All(OvA) algorithm with KNN improved baseline accuracy in a minimal percent. This suggesting that combining OvA with KNN and features such as color, texture, size and asymmetry enchances model performance and accuracy.**

Our research highlights the potential of machine learning in dermatological diagnostics, demonstrating the successful integration of computer vision and medical diagnostics. Further investigations are needed to develop more tools for dermatologists and expand applications to other medical and related fields

***Index Terms*—Machine learning, skin cancer, supervised learning, computer vision, kNN, One vs All, PCA, Region of Interest, computed asymmetry.**

I. Introduction

Sight is inherent to our being, but we are not the only ones who can analyze our surroundings. We can provide of a little bit of our capacities to the computers by programming and testing models that are able to extract information of the pictures and process it in order to manipulate it and satisfy any task. One of the main goals of the computer vision according to the university of San Diego [University of San Diego(2024)] is to enable computers to analyze and understand visual data in much the same way that human brains and eyes do, and to use this understanding to make intelligent decisions based on that data. In this paper we are gonna dive into the application of this technology on the medical field, specifically into the detection of skin cancer, to achieve that we took a dataset with 10015 cancer lesions and images [Mader(2018)], and use the information to train our model, reporting the results on this paper.

A. Motivation

Machine learning and computer vision are tools that, if used in the right direction, can give us a lot of possibilities in different areas, such as security, medicine, creativity, etc. It is our responsibility as AI engineering students to provide solutions to the problems that mankind has endured for many years. The increasing incidence of skin cancer, and the critical need to find early and accurate diagnosis to treat this disease are important problems in our time. This project can serve us as a pathway to gain comprehension, knowledge and practical experience in these technologies, so we can apply them in various complex problems and generate solutions for this and other issues in life.

B. Aim

K Scott Mader's Skin Cancer MNIST: HAM10000, is a collection of dermatoscopic images of pigmented lesions available in the web. Based on this datasets, the primary aim of this project is to develop a machine learning model capable of accurately classify skin lesions that may be cancerous, working with images of 7 different pigmented lesions and it's data, available in the HAM1000 dataset. We want to test and compare different methodologies for solving this problem using different approaches as one vs all or region of interest cropping. And the other aim of this project is to extract the asymmetry data of every image, so we can relate it to the 7 different labels and observe their features. We will improve alongside our artificial intelligence model, ultimately getting to identify the most optimal formula for achieving the best diagnostic results.

C. Research questions

1) How is the performance of the model affected when applied over extracting RGB information from the original images vs from the ROI (Region Of Interest) of the images?
2) How can other algorithms like One vs All make a difference comparing the accuracy of only using KNN? Does the accuracy improve with the ROI preprocessing?
3) How does the computed asymmetry of skin lesions relate to the given ground-truth of HAM1000?

D. Organization of the rest of the report

The rest of the report is divided in scientific background, with the related works in the field, the explanation of the skin image dataset, the proposed methods for the skin image classification, the experimental setup with some baseline approaches, validation metrics and explanation of the implementations, and last but not least, the results, discussion and conclusions of the project.

II. Scientific Background

A. Related works

Talking about the related works we can find a lot of information about skin cancer or skin injuries, our most advanced technology can help us to improve the detection of some illnesses, but before that we need to train our models in order to make a powerful tool that allow us to take a step forward on the medical field (or any other field). The harder we work to achieve more knowledge of our models in order to be more prepared is the more benefit we get, at the end we are the principal benefactors of this work, we can detect sooner if somebody is sick and act properly and efficiently. Our investigation lead us to the following related works on skin cancer because we know that this terrible illness can appear on any part of the body, and its crucial to have and investigate more ways to prevent or treat skin cancer.

1) Detection of skin diseases from Dermoscopy image using the combination of convolutional neural network and one-versus-all: [Polat & Koc(2020)] This work show a massive example of what we can do with the dataset given, the goal with the HAM10000 or any other dataset that works on skin cancer detection is to achieve the correct classification of the images with the corresponding labels, the authors work with CNN (Convolutional Neural Network) obtaining a 77% classification accuracy only with CNN, and the combination of CNN and one-versus-all approach achieved 92.9% accuracy. The obtained results have shown that the proposed method by the authors is very promising in the classification of skin disease from Dermoscopy images. Pro's about this study is the high accuracy that was obtained combining CNN with One-versus-all in classifying various skin diseases, exploiting the strengths of this combined model approach, but a negative point is that training CNNs and implementing One-Versus-All can be computationally expensive and time-consuming, on the report its mentioned that the results taken 4 hours to be reached.

2) AI Techniques of Dermoscopy Image Analysis for the Early Detection of Skin Lesions Based on Combined CNN Features: [Olayah et al.(2023)Olayah, Senan, Ahmed & Awaji] This work inspire us on looking for more different ways to compare the performance of our model by extracting the features in different ways, in the mentioned work the authors worked with a different dataset that contains 25,331 images from the HAM10000 and BCN_20000 datasets. In this work also the authors implemented the Geometric Active Contour that allowed them to segment and store images with only the Region Of Interest, extracting the affected zone of the skin. Positive things about this work are the research which includes a comparative analysis of different AI techniques, providing insights into the strengths and weaknesses of various approaches. Also Combining features from different CNN architectures to improve classification accuracy and robustness. But in the same way a issue could be that combining features from multiple CNNs increases the complexity of the model, making it harder to implement and optimize and requiring significant computational power and memory.

3) Synthesis: Both studies leverage CNNs to effectively classify skin diseases from dermoscopy images, the models demonstrate high accuracy, specificity, and sensitivity, which are critical for medical diagnostics. But also both studies tend to increase model complexity, which can be challenging in terms of implementation and optimization.

III. Skin image dataset

The dataset we're working with is the K Scott Mader's Skin Cancer MNIST: HAM10000, that stands for Human Against Machine 10000 training images. [Mader(2018)] To reach a better viability of the ground-truth of the dataset our research lead us to a metric of unsupervised learning and although we work with supervised learning, this metric called Clustering Silhouette is used to provide valuable insights in a supervised learning project, particularly during the exploratory data analysis (EDA), the Silhouette Score can help us to assess how well the data points cluster naturally, by clustering the data and calculating the Silhouette Score, we can gauge how distinct the different classes are from each other. Higher scores might indicate that classes are well-separated, which could imply that the features from the dataset are effective at distinguishing between classes. The dataset collected dermatoscopic images from different populations, the final dataset consist of 10015 dermatoscopic images which will serve us as a training set. Cases include 7 diagnostic categories of pigmented lesions classified as can be seen on the Figure 2. Having a ground-truth well defined is a crucial consideration for the correct development of the model, because of that it can be implemented different metrics even if they are not correspond to supervised learning, in this case we implement the clustering silhouette analysis to determine how well the data has been clustered, this can be done by measuring how similar each point is to other points in its own cluster compared to points in other clusters, assessing cohesion [Shutaywi & Kachouie(2021)]. By using the libraries from scikit-learn we can calculate silhouette coefficients for each point in the clusters and compute the average silhouette score for the entire clustering. [Learn(2017)]. The visualization of this clustering is on the Figure 1.

We think that in this particular case, the presence of 'nv' in different parts of the silhouette can generate confusion or misinterpretation problems with the images.

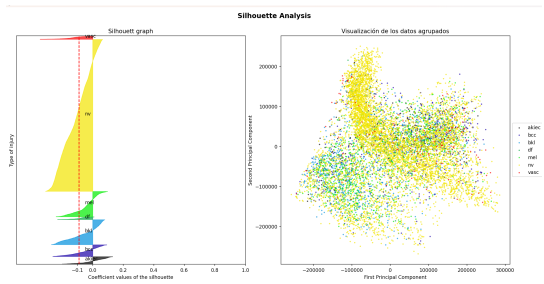

Fig. 1: Clustering Silhouette of the dataset.

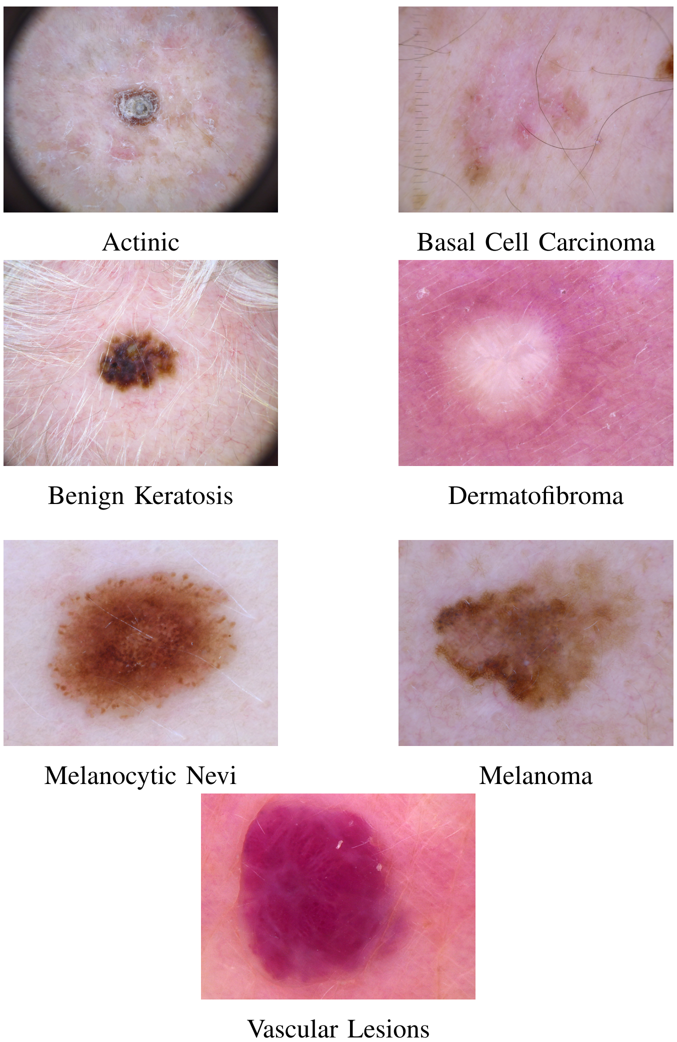

Fig. 2: Samples of Skin Lesions

IV. Methodology: Proposed methods for Skin image classification

A. Region Of Interest (ROI) Extraction

In order to improve and compare the performance of the model, we made an improvement to the form that we extract information of the pictures, implementing an algorithm called Geometric Active Contour, which allow us to extract what we determinate as Region Of Interest(ROI), that is only the

Dx	Dx type	Age	Sex	Localization
nv 67% mel 11% Other 22%	histo 53% follow up 37% Other 10%	45 : 13.04% 40 : 9.89% 35 : 7.56% 30 : 4.66%	Male: 54% Female: 45% Other: 1%	Back: 22% Lower extremity: 21%

TABLE I: Describing some features of the dataset

area affected by the injury. So it passes from a "raw image" to a image that is only remarking the zone with the injury, implementing it we made another folder with the same 10015 images but only showing the injury, by doing this we want to compare the performance of our model using RGB extraction with the normal images versus using RGB extraction of only the affected area. The steps that we follow are described on the following sections.

- Hair removal:
  The first step in order to detect the contours is to remove the hair in the image. This is only for improving the contour detection, the final image still has hair on it, because there are other important attributes of the lesions that are removed with the hair when doing this process. We use Skimage package to convert the image to grayscale and applying a "blackhat" filter to find hair. Then we create a binary mask of the hair, and use the OpenCV's library inpaint function to fill in the hair areas.
- Lesion contour detection:
  The lesion contour is obtained by using a function that employs two different approaches depending on whether a border is detected in the image. The lesions of the images with this black border in the dataset seen to be tiny and located in the center, so it's necessary to make this distinction in order to detect the contours.

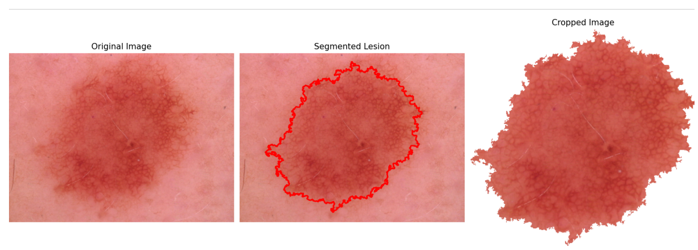

Fig. 3: Lesion contour detection in a normal image

For a normal image, the procedure is to convert it to grayscale and apply Otsu's thresholding to binarize the image. Then we find the contours by using the findcontours function of skimage, and select the largest contour. For a image with black border, we focus on the region of the center of the image and apply Otsu's thresholding to this. Morphological operations are performed to clean up the binary image, and then we find the contours using the same method as the normal image. Finally, we adjust the contour coordinates to the full image space.

This procedure can be plotted, and it looks as in Fig. 1 and Fig. 2.

Fig. 4: Lesion contour detection in an image with a black border

- Lesion cropping:
  First, we create a binary mask using the lesion contour, and create an RGBA image with a transparent background. The program uses the mask to make the background outside the lesion transparent, and finds the bounding box of the lesion. It crops the RGBA image to this bounding box and save the cropped image as a PNG.
- Feature extraction:
  We apply the previous process to the 10015 images in the dataset and store them in a folder. It takes around 2 hours and a half. We read the CSV dataset file using pandas, and iterate through the second column which contains the image IDs. For each one of this, the program constructs the full path to the corresponding PNG image, and we extract it's rgb features one by one, by creating a 24 element array where a 8 bins histogram counts, ranging from 0 to 255 are stored. The histogram vector for each image is appended to a list, and after processing all images, it is converted to a numpy array and saved using np.save().

The output of this stage is a list with each image cropped image histogram vectors.

B. Asymmetry extraction

As we ventured deeper into our project, we realized that while traditional features such as color and texture offer valuable insights, they often miss a crucial element asymmetry.

Asymmetry in a skin lesion is a significant indicator, providing vital information that can enhance our understanding and analysis. Thus, we decided to incorporate asymmetry into our study to examine its relationship with the ground truth and gain a more comprehensive understanding of the data.

In a review of skin lesion classification and detection methods, it was highlighted that asymmetry, along with other features, plays a critical role in the accurate diagnosis of skin conditions. The study emphasizes the importance of incorporating various morphological features, including asymmetry, to improve diagnostic accuracy and reliability in machine learning models for skin lesion analysis [Debelee(2023)]

- Asymmetry analysis: This code performs asymmetry analysis on a single skin lesion image. It begins by importing essential libraries such as numpy for numerical operations, matplotlib.pyplot for visualization, and skimage for image processing.
  The code computes asymmetry indices by splitting a binary mask of the lesion into vertical and horizontal halves, calculating differences between these halves, and normalizing by total pixel counts. Then images are loaded as an PNG image, that are later transformed into a floating-point format, that generates a binary mask based on image transparency (for RGBA) or grayscale thresholding (for RGB). It then applies the asymmetry calculation function to compute and visualize vertical and horizontal asymmetry indices.
  Additionally the program includes a visualization part for a better understanding of the results that displays the original image, the binary mask, and visual comparisons of the left-right and top bottom halves with red dividing lines indicating the points of symmetry. Finally, the calculated asymmetry indices are returned and printed, providing a clear numerical indication of the asymmetry present in the lesion.

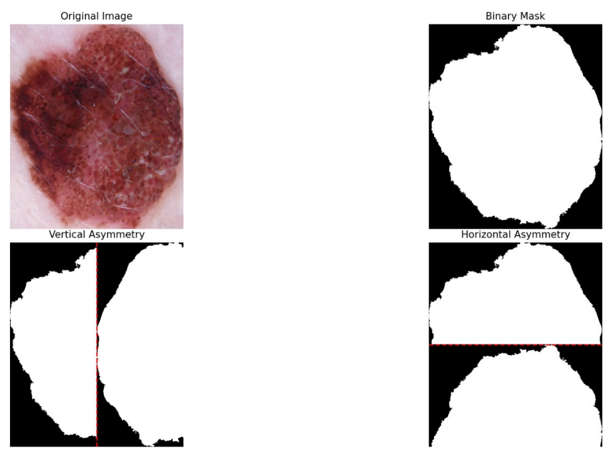

Fig. 5: Asymmetry analysis

Asymmetry Values
Vertical Asymmetry: 0.1052
Horizontal Asymmetry: 0.0658

Fig. 6: Table showing asymmetry values for this lesion.

- Analysis of all the images: Now we need to make the analysis for all the images, the program also process images and calculates the asymmetry but the main difference is that now the program reads both things, the image metadata from a CSV file using pandas, and iterates over each image in the folder, computing the asymmetry and storing the results (image ID, classification, vertical asymmetry, and horizontal asymmetry) in a new CSV file. Finally, the results are saved into asymmetry results.csv, which contains the computed asymmetry indices for each image.

- Asymmetry average : Finally with the results from the CVS file we use pandas for efficient data manipulation. The data is grouped by the classification column to aggregate and calculate the average values of vertical and horizontal asymmetry for each category. This proccess is easier to implement using pandas "groupby" function, which allows us to summarize and analyze the asymmetry data effectively. The resulting averages are then renamed and displayed in the console. Additionally, these summarized results are saved into a new CSV file named "resultados promedios.csv"

C. RGB extraction

This RGB extraction method serves as a crucial first step in our image classification pipeline. By reducing each image to a concise 24-element vector, we capture essential color information while signifi- cantly reducing the dimensionality of our data. This reduction is particularly beneficial for the subsequent PCA step, allowing us to further refine our feature set while retaining the most color information from our dermatological images. The color histograms provide valuable insights into the over- all color distribution of skin lesions, which can be indicative of various dermatological conditions. By focusing on this color information, we lay the groundwork for more sophisticated analysis in the later stages of our deep learning classification system

D. PCA

PCA allow us to visualize the 10015x24 dataset rgb features in a 10015x2 matrix, making it possible to visualize it on a 2D plot. We also store it in a NumPy file and use it in the KNN to compare results.

In first place, we load a NumPy array with the rgb features of the dataset, and categories from the CSV File. PCA is initialized with 2 components using the sklearn.decomposition library, and then is applied to the rgb features vectors. Now it can be stored in a NumPy file. For the 2D visualization, a scatter plot is set up using matplotlib and assigning colors to each category. For each category, belonging points are identified and plotted on the scatter plot with a specific color.

E. Stratifed K-Fold Cross-Validation Sets

In order to properly train the model, we use stratified training sets and test sets of the dataset indices. We start by inputting the different labels in the order they appear in the csv, with their respective counts. A pandas DataFrame is created with an index column and a label column. The dataset is first split into training (80 percent) and test (20 percent) sets using traintestsplit. This split is stratified. Then, StratifiedKFold is initialized with nsplits = 5, shuffle = True, and a random state.

The split method is used to generate 5 different train-validation splits of training data. It generates a training and a validation subset for each fold, and these are appended to different lists that are stored in a dicionary called indicesdata, that also includes the test set. Finally, the indices for training, validation and test are saved to separate JSON files.

F. Image classification - ML model

Because of the need of use all type of information available on the HAM10000 we realized that it is possible to extract a histogram of the color of the images with the injuries, which will allow us to extract features in order to have more information for our model.

After extracting the RGB information we reduce the amount of information with a PCA (Principal Component Analysis) to reduce the size of the matrix that we are managing from 10,000X24 to 10,000X2, and showing it with a plot function to see how it seems with the 7 categories of injuries that we are analyzing. Then we classify the information doing a session of training and testing with the K-Fold Cross-validation algorithm, doing the train 5 times and obtaining several results.

We implement the K Nearest Neighbor algorithm for knowing our performance, with graphs that show us the accuracy of the model, F1-Score and more metrics with an average of the 5 splits, more details will be explained on the section implementation details. Is employed to classify the data points based on their proximity to the other points in the dataset, following the idea that similar data points tend to have the same label or values. To make predictions kNN needs to calculate the distance between input data points (from the dataset) and the training examples that were stored using a distance metric like euclidean distance. First thing first is importing libraries, which will be more detailed on the section implementation details, load the data (CSV), establishing the K = 3, which means that to make a prediction, it considers the 3 nearest neighbors, the label that appears with more frequency between this 3 points will be the assigned for the point, and in a case where it can been 3 different labels the chosen one will be the class of the nearest neighbor, the closeness will be calculated with euclidean distance. Afterwards with the index extracted on the Cross Validation and loaded from a JSON file, we can access to the train, validation and test sets indices in order to achieve the cross validation.

Asymmetry is one of the key criteria in the ABCDE rule (Asymmetry, Border irregularity, Color variation, Diameter, and Evolving) used by dermatologists to assess skin lesions for potential malignancy, particularly melanoma. Malignant lesions are often asymmetrical, whereas benign ones tend to be more symmetrical, because of that we analyze the asymmetry of the injuries of the dataset and compare it with the given ground-truth, asymmetrical patterns in skin lesions can indicate irregular growth and structural changes within the tissue, which are characteristic of malignancies. Identifying these patterns through feature extraction can significantly enhance the accuracy of automated diagnostic systems.

By training the system using a KNN with the One vs All algorithm, we'll be dividing the KNN in seven binary trainers instead of just one trainer with seven classifications. It has been proven by the authors of *Detection of skin diseases from dermoscopy image using the combination of convolutional* *neural network and one-versus-all.* [Polat & Koc(2020)] that these method gives better and more accurate results, improving them in 15%, so we hope that it improves the performance of the model accuracy. The One-vs-All approach adapts KNN for multi-class problems by creating multiple binary classifiers, for each class a KNN classifier is trained to distinguish that class from all others, to make predictions, each classifier calculates the distance between input data points and the stored training examples using a distance metric like Euclidean distance. The process its almost the same as the used on kNN but here the final label is determined by combining predictions from all binary classifiers, typically choosing the class with the most confident prediction. Each binary classifier focuses on distinguishing a specific class. The visualizations and validation metrics are the same, more details about validation metrics will be on the mentioned section.

V. Experimental Setup

A. Baseline approaches

There are basic approaches that we can compare our job to. One of them may be extracting the rgb features from the 10015 images, preparing cross-validation sets [Prusty et al.(2022)Prusty, Patnaik & Dash] and training a model with KNN [Jolly(2018)], establishing the number of K-Nearest Neighbours as 3.

The results of this approach are displayed in table II and table III, as well as figure 5.

TABLE II: Validation and Test Results by Split

Split	Validation Accuracy	Validation F1-score
Split 1	0.6831	0.6841
Split 2	0.6912	0.6944
Split 3	0.6910	0.6864
Split 4	0.6785	0.6798
Split 5	0.6879	0.6916

Split	Test Accuracy	Test F1-score
Split 1	0.6925	0.6981
Split 2	0.6995	0.7038
Split 3	0.6895	0.6950
Split 4	0.6880	0.6924
Split 5	0.6935	0.6970

	Value
Accuracy	0.69
F1 Score	0.70
Precision	0.71

TABLE III: Test Mean Values

B. Validation metrics

In supervised learning several key validation metrics are commonly used to evaluate the performance of a model, these metrics like accuracy, precision, and F1 score, provide different insights into the model's performance [Hossin & Sulaiman(2015)]. In order to measure our results we employed different metrics to evaluate our models performance, trying to made them in the most clear and understandable way:

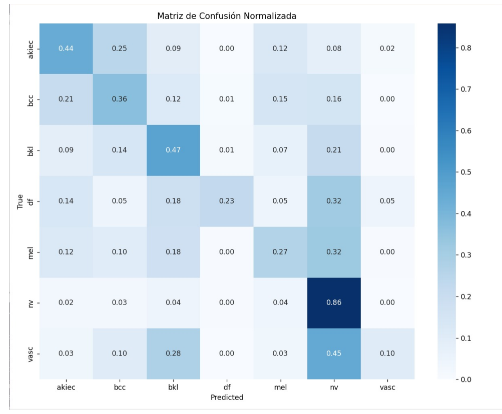

Fig. 7: Baseline Confusion Matrix

1) Normalized confusion matrix: A normalized confusion matrix is a valuable tool in evaluating the performance of a classification model. Unlike a standard confusion matrix that shows the raw counts of true positives, true negatives, false positives, and false negatives, a normalized confusion matrix presents these counts as proportions, which can be particularly useful for comparing models or understanding performance across classes with imbalanced datasets. [Jolly(2018)]

2) Accuracy: Accuracy is the ratio of correctly predicted instances to the total instances. It is a straightforward metric that indicates how often the model is correct overall. However, accuracy can be misleading, especially in cases of imbalanced datasets, where one class may dominate the other. [Sokolova & Lapalme(2009)]

3) Precision: Precision is the ratio of correctly predicted positive observations to the total predicted positives. It answers the question: "Of all the instances that were predicted as positive, how many were actually positive?" Precision is crucial when the cost of false positives is high. [Sokolova & Lapalme(2009)]

4) F1-Score: The F1-score is the harmonic mean of precision and recall, providing a single metric that balances both. It is particularly useful when you need to account for both precision and recall in the presence of class imbalance [Sokolova & Lapalme(2009)]. The F1 score is calculated as:

$$\text{F1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \quad (1)$$

C. Implementation details

To read or use the respective codes used in this model, you can go to https://github.com/Adlerero/Codes-For-Machine-Learning-Skin-Cancer-Model.

1) RGB Extraction: Python provides of the scikit image library, that allow us to manipulate different image features as this RGB extraction. In order to implement it, we need libraries as imageio.v3 to read the image, matplotlib to make a visualization of the RGB histogram, numpy for numerical operations or generating the numpy vector file, and the previously mentioned skimage.

2) Region Of Interest (ROI) extraction [Polat & Koc(2020)]: The code is made in python, because of its libraries for image manipulation. The ones used are numpy, for numerical operations, skimage for applying filters and other image operations, cv2 for the inpaint function, skimage.draw for drawing the contour polygon, matplotlib for saving the image, os for file system operations as generating a folder for the cropped images, imageio.v3 for reading image files, and pandas for handling CSV files.

3) Assymetry extraction: The code is made in python, because of its libraries for image manipulation. Again, scikit-image can perform symmetry operations, so we use skimage, matplotlib for the visualization of the asymmetry, numpy for numerical operations, os to access or creating new files in the system, and pandas to read csv files and also generating them.

To read the asymmetry, we need to know that closeness to 0 means a more symmetric image, while closeness to 1 means a more assymetric image.

4) PCA: [Rodrigo(2020)] The libraries used are numpy, for efficient numeri- cal operations, pandas for data manipulation and analysis, sklearn.decomposition, for machine learning features and the PCA implementation, and mat- plotlib to make a visualization of the PCA.

This scikitlearn library is very important, because with just 2 lines of code you can implement the PCA with all the operations that are needed for it. Reducting a 10015x24 matrix to a 10015x2 matrix is not easy and requieres a lot of operations. [Maćkiewicz & Ratajczak(1993)]

However, the numpy file we used for training the model was the larger one (10015x24), because it gave much better results in the baseline approach.

5) K-Fold Cross-validation sets: [Learn(2019)] The libraries used are numpy, for efficient numerical operations, pandas for data manipulation and analysis, sklearn.modelselection importing StratifiedKFold, traintestsplit, that provides tools for model selection and evaluation, and json, used for saving data in JSON format. The skf.split method is used to divide the training set in 5 different folds.

6) K-Nearest Neighbors: [learn Developers(2024b)] The implementation is done on python, it uses libraries including numpy for efficient numerical operations, pandas for data manipulation, sklearn.neighbors for the k-NN algorithm, sklearn.preprocessing for data scaling, and matplotlib and seaborn for visualization. Initially, the dataset is loaded from our CSV file containing the metadata and a NumPy file containing preprocessed RGB feature vectors, the RGB feature vectors are combined with the metadata features to form a comprehensive feature matrix. To ensure robust evaluation, the dataset is split into training, validation, and test sets using indices stored in JSON files, these indices define stratified splits, ensuring that each subset maintains the same distribution of classes as the original dataset.

The results of each fold are stored in lists called validation scores and test scores, then on the cross validation we divide the data, train the model and evaluate the validation and test set, true predictions and true labels were saved on the last fold.

The final evaluation is performed on the test set. Various metrics such as accuracy, F1-score, precision, and recall are computed to indicate the classifier's performance. Additionally, using the matplotlib.pyplot library we make plot (plt) figures to show a graph of precision per fold and a confusion matrix that is plotted to visualize the classifier's performance across different classes, finally we printed the validation and test accuracy alongside the F1-Score, precision, recall and the average of accuracy and F1-Score.

7) K-Nearest Neighbors with One vs All: [Varpa et al.(2011)Varpa, Joutsijoki, Iltanen & Juhola] On implementation we follow almost the same line that k-NN except that after scaling the features we need to wrap the k-NN classifier with OneVsRestClassifier to handle multi-class classification, this involves applying the One-vs-Rest strategy, where we train multiple binary classifiers (one for each class) and use them to handle a multi-class classification problem. This is done before initialize the lists to store the results, we create an object from the library sklearn.multiclass [learn Developers(2024a)], we use the class OneVsRestClassifier. We train the model with this created object, using the method fit taking parameters of the train indices. Then we perform cross-validation using the provided training and validation folds, later we train the model on training data and evaluate on validation and test data, we store and calculate accuracy for each fold to be shown later, the rest implementation its for the plot graphs and it follows the same line as kNN.

VI. Results

A. KNN using One vs All

The baseline approach accuracy was improved by 2 percent by using KNN with One vs All algorithm. These are the results (Check Table IV, Table V and Fig. 8 for Confusion Matrix):

TABLE IV: One vs All Validation and Test Results by Split

Split	Validation Accuracy	Validation F1-score
Split 1	0.6993	0.6741
Split 2	0.6999	0.6779
Split 3	0.7085	0.6794
Split 4	0.6991	0.6766
Split 5	0.6954	0.6724

Split	Test Accuracy	Test F1-score
Split 1	0.7114	0.6936
Split 2	0.7099	0.6873
Split 3	0.7129	0.6935
Split 4	0.7124	0.6905
Split 5	0.7084	0.6863

We can observe that both Fig. 5 and Fig. 8 have very good performances for 'nv', with high correct classification rates.

	Value
Accuracy	0.71
F1 Score	0.69
Precision	0.67

TABLE V: One vs All Test Mean Values

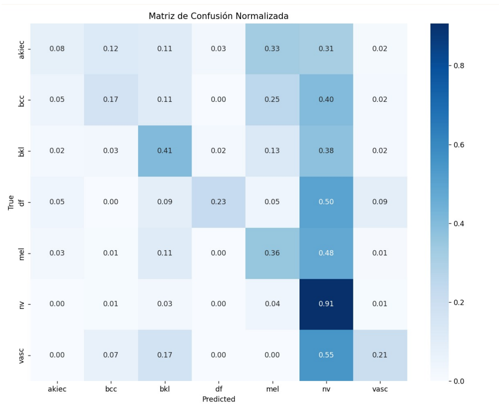

Fig. 8: One vs All Confusion Matrix

However, both matrices show poor performance for 'df'. This is particularly interesting, because while 'nv' represents 66.9 percent of the total samples, 'df' has only 1.1 percent of them. Other topic to add is that in both figures, several classes are frequently confused with 'nv'.

The cause of all this problems may be the balance of the dataset, and we think that a more balanced dataset could offer better results.

B. Region Of Interest extraction

On the other hand, the results when using Region Of Interest extraction improve the accuracy of the baseline results (by just 1 percent), but not the KNN with One vs All results (check Table VI, VII, VIII, IX and Fig. 9, Fig. 10 for Confusion Matrix):

TABLE VI: ROI Extraction Validation and Test Results by Split

Split	Validation Accuracy	Validation F1-score
Split 1	0.6856	0.6840
Split 2	0.7087	0.7039
Split 3	0.6998	0.6970
Split 4	0.6904	0.6900
Split 5	0.6979	0.6936

Split	Test Accuracy	Test F1-score
Split 1	0.6990	0.6992
Split 2	0.6900	0.6845
Split 3	0.7034	0.7023
Split 4	0.6960	0.6950
Split 5	0.6980	0.6981

The same problem remains. 'nv' has the best performance, but other classification performances could improve. On the other hand, we can see that while some classification perfor-

	Value
Accuracy	0.70
F1 Score	0.70
Precision	0.70

TABLE VII: ROI Extraction Test Mean Values

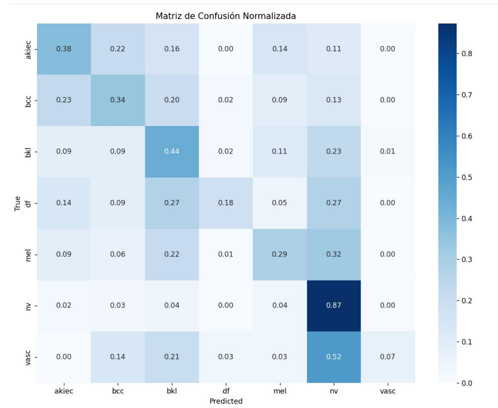

Fig. 9: ROI Extraction Confusion Matrix

mances like 'akiec' and 'bcc' improve a lot, principally in figure 9, other ones have a low downgrade.

C. Asymmetry

These results support the importance of including asymmetry analysis in skin lesion classification models. They also suggest that while asymmetry is a valuable indicator, it should be used in conjunction with other features for accurate diagnosis, as some malignant lesions may not always present high asymmetry.

VII. Discussion

A. Answer to research questions

- How is the performance of the model affected when applied over extracting RGB information from the original images vs from the ROI (Region Of Interest) of the images?

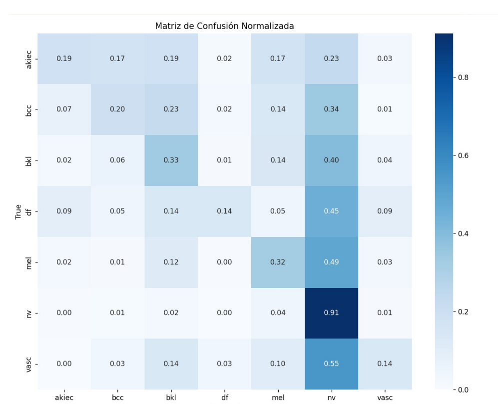

Fig. 10: ROI Extraction using One vs All Confusion Matrix

TABLE VIII: ROI Extraction Using OvA Validation and Test Results by Split

Split	Validation Accuracy	Validation F1-score
Split 1	0.6974	0.6740
Split 2	0.7099	0.6865
Split 3	0.7029	0.6780
Split 4	0.6985	0.6740
Split 5	0.7085	0.6873

Split	Test Accuracy	Test F1-score
Split 1	0.7059	0.6835
Split 2	0.6965	0.6676
Split 3	0.7159	0.6934
Split 4	0.7084	0.6860
Split 5	0.7034	0.6823

	Value
Accuracy	0.70
F1 Score	0.68
Precision	0.67

TABLE IX: ROI Extraction using One vs All Test Mean Values

Our hypothesis lead us to the idea that maybe with more specific information of the RGB image only with the injury, the performance of the model could be improved, but to reach this hypothesis we also need to extract the ROI from every image o the dataset, resulting on a new folder with this images that took almost 3 hours to be complete. Afterwards the model using kNN and information extracted from the ROI showed a small improvement, having an accuracy of 0.70 against an accuracy of 0.69 with the normal images, but when we implement kNN with OvA algorithm it shows a performance difference, having an accuracy of 0.71 with the normal images, against, an accuracy of 0.70 with kNN OvA and the images with the ROI. This performance showed that when the model used the RGB information extracted with the ROI it can be possible a little improvement, but this is only if the model is using kNN, cause when kNN-OvA is implemented the better result is with the original images. Obviously the images with ROI cropped are not perfectly done, although the main goal was to do it, it comes with several complications of every specific sample, like the shape of the injury, the disproportionality, etcetera, but after all the complications it achieves promising results.

- How can other algorithms like One vs All make a difference comparing the accuracy of only using KNN? Does the accuracy improve with the ROI preprocessing?
  The One vs All logic proved to work very well by improving the accuracy in different works we read [Varpa et al.(2011)Varpa, Joutsijoki, Iltanen & Juhola] [Olayah et al.(2023)Olayah, Senan, Ahmed & Awaji]. The hypothesis was that, by dividing the kNN with 7 classifications into 7 kNN's with binary classifications, the model can better handle multi-class problems, being easier to manage and obtaining better performance and accuracy results. Our results proved this hypothesis, with the model accuracy being improved by 2 percent. Nonetheless, when it comes to preproccesed images with the ROI extraction, the accuracy between applying kNN and kNN with OvA was the same. This suggests that the ROI model may have reached its performance ceiling with the available features, and the added complexity of OvA might not be justified for this particular problem. This performance showed that when the model is only using kNN and ROI images it can be possible a little improvement, but if we combine OvA with kNN its better to use the normal images instead of the images with the ROI cropped.
  We also need to consider the unbalance of the given ground truth, having almost 70 percent of just one classification. The model may improve if we can analyze a balanced dataset with a similar percentage of each label.

Classification	Average Vertical Asymmetry	Average Horizontal Asymmetry
akiec	0.446344	0.429835
bcc	0.386680	0.381537
bkl	0.344958	0.340884
df	0.241806	0.235243
mel	0.285258	0.279943
nv	0.210693	0.207519
vasc	0.213695	0.210074

TABLE X: Average Vertical and Horizontal Asymmetry by Classification

- How does the computed asymmetry relate to the given ground-truth of HAM1000?
  The computed asymmetry relates to the ground-truth in how we could observe how the asymmetry values vary across different types of skin lesions. This analysis revealed that different lesion types exhibit distinct asymmetry patterns that were specific to each classification.
  We also observed that although melanomas tend to have higher asymmetry on average, our analysis indicates that this indicator alone should not be used to accurately classify each type of lesion. It is essential to include other key features due to potential bias in the model, as asymmetry can vary significantly. For example, in our model, the lesion type with the highest average asymmetry was akiec, with a vertical asymmetry of 0.446344 and horizontal asymmetry of 0.429835, compared to melanoma's vertical asymmetry of 0.285258 and horizontal asymmetry of 0.279943.
  By correlating asymmetry with the known diagnoses in the HAM1000 dataset alongside other key features, we can improve the accuracy and reliability of our models.

B. Other topics

1) Accountability in the use of machine learning models for skin lesion detection: The integration of machine learning models in the diagnosis of skin cancer raises important questions about accountability. While these models can help us in the early detection and accurate diagnosis they also prompt the question: who bears responsibility if something goes wrong? In this context, accountability does not relies on one individual or entity but is shared among multiple people involved in the methodology. From model developers and healthcare professionals using them, to those responsible for regulating their ethical and safe implementation, each plays a crucial role. This shared accountability underscores the need for clear regulations, rigorous analysis, and ethical, transparent use of the technology to ensure reliable and safe outcome in medical diagnosis.

C. Limitations of your work

Probably the nearest limitation its the time, although we will be enchanted to dedicate more time in order to reach better results, we are limited of time. Other limitation may be the lack of information about some types of cancers, cause some information its limited for medical purposes. Working on the model and on the feature extraction, specifically on the ROI, we face up a problem, and it is the fact that in order to compare the performance of the model with RGB extraction of the original images from the dataset, versus, the performance of the model extracting RGB information of only the ROI, we had to create a new folder with the 10015 images cropping only the zone of the injury, this task took almost 3 hours to be done and obviously, the process of completing this task might be different but it will depend on the resources of every computer.

D. Future work

Our work its not done yet, because we can always improve in any way and although we collect different results, our inherent curiosity will always lead us to try to accomplish the main goal, make a model capable of classify images with the best accuracy possible, by implementing new features like Neural Networks, a very complex topic that can be very helpful on the related work with models with supervised learning.

Other possibility is to work with the RGBA features of the PNG cropped ROI images as well as other RGB features, so the transparency can also give us information and maybe improve the accuracy of the model.

Work with a multi-modal data integration its also another research that could be very helpful, incorporating additional data types such as patient history, genetic information, and dermoscopic images alongside the current RGB data, potentially improving diagnostic accuracy but trying to no over-fit the model.

For a future research it will be very interesting dive more into this topics or even work on expanding the dataset, cause it need to be more diversity of the samples, in this way the model can be trained to detect injuries on any kind of skin; as well as working with a more balanced model, that has more similar percentages for each category instead of more than 60 percent for just one (nv in this case).

VIII. CONCLUSIONS

In this project, we successfully leveraged a skin cancer image dataset to explore the potential of machine learning in dermatological diagnostics. We learned about RGB extraction to capture relevant color information from this skin lesion images, and an important approach to detect potential malignancies; the application and importance of Principal Component Analysis (PCA), to reduce the dimensionality of our feature space while retaining essential information, with the possibility of visualizing the data; the use of Stratified K-Fold Cross-Validation ensured a more reliable assessment of our model's performance, preventing potential imbalances; the K-Nearest Neighbors (KNN) proved to be a viable algorithm for this supervised learning classification task, offering effectiveness and good accuracy for this image-based diagnostics.

The application of the Silhouette method was also crucial for verifying the reliability of the given ground truth labels, enhancing our confidence in the dataset's quality and subsequent evaluations. And the interpretation of confusion matrix, F1-Score and other types of results helped us to evaluate our models and determine their efficiency.

Our research successfully demonstrated improved accuracy in skin lesion classification when using the One vs All algorithm. The incorporation of asymmetry analysis proved pivotal, aligning closely with ground truth data an enhancing the labels classification to distinguish between benign and malignant lesions. The performance evaluation revealed just little enhancements when utilizing Region of Interest (ROI) images, so we will keep investigating how to improve the model by using this features.

This project demonstrated the successful integration of computer vision, machine learning and medical diagnostics given by the dataset, highlighting the potential for AI in healthcare. Although much further investigation is needed, this work shows the possibility of developing assistive tools for dermatologists and the possible detection of skin cancer with AI, and the possibility of developing other models that belong to other fields as objects detection, security and surveillance, etc.

REFERENCES

[Debelee(2023)] Debelee, T. G. (2023). Skin lesion classification and detection using machine learning techniques: A systematic review. *Diagnostics*, *13*, 3147.

[learn Developers(2024a)] learn Developers, S. (2024a). Onevsrestclassifier.

[learn Developers(2024b)] learn Developers, S. (2024b). sklearn.neighbors.kneighborsclassifier.

[Hossin & Sulaiman(2015)] Hossin, M., & Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International Journal of Data Mining & Knowledge Management Process*, *5*, 1–11.

[Jolly(2018)] Jolly, K. (2018). *Machine learning with scikit-learn quick start guide: classification, regression, and clustering techniques in Python*. Packt Publishing Ltd.

[Learn(2017)] Learn, S. (2017). Selecting the number of clusters with silhouette analysis on kmeans clustering.

[Learn(2019)] Learn, S. (2019). Cross-validation: evaluating estimator performance.

[Maćkiewicz & Ratajczak(1993)] Maćkiewicz, A., & Ratajczak, W. (1993). Principal components analysis (pca). *Computers & Geosciences*, *19*, 303–342.

[Mader(2018)] Mader, K. S. (2018). Skin cancer mnist: Ham10000. *Skin Cancer*, .

[Olayah et al.(2023)Olayah, Senan, Ahmed & Awaji] Olayah, F., Senan, E. M., Ahmed, I. A., & Awaji, B. (2023). Ai techniques of dermoscopy image analysis for the early detection of skin lesions based on combined cnn features. *Diagnostics*, *13*, 1314.

[Polat & Koc(2020)] Polat, K., & Koc, K. O. (2020). Detection of skin diseases from dermoscopy image using the combination of convolutional neural network and one-versus-all. *Journal of Artificial Intelligence and Systems*, *2*, 80–97.

[Prusty et al.(2022)Prusty, Patnaik & Dash] Prusty, S., Patnaik, S., & Dash, S. K. (2022). Skcv: Stratified k-fold cross-validation on ml classifiers for predicting cervical cancer. *Frontiers in Nanotechnology*, *4*, 972421.

[Rodrigo(2020)] Rodrigo, J. A. (2020). Reproducibilidad de las componentes pca.

[Shutaywi & Kachouie(2021)] Shutaywi, M., & Kachouie, N. N. (2021). Silhouette analysis for performance evaluation in machine learning with applications to clustering. *Entropy*, *23*, 759.

[Sokolova & Lapalme(2009)] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, *45*, 427–437.

[University of San Diego(2024)] University of San Diego (2024). Introduction to computer vision. Accessed: 2024-07-08.

[Varpa et al.(2011)Varpa, Joutsijoki, Iltanen & Juhola] Varpa, K., Joutsijoki, H., Iltanen, K., & Juhola, M. (2011). Applying one-vs-one and one-vs-all classifiers in k-nearest neighbour method and support vector machines to an otoneurological multi-class problem. In *User centred networked health care* (pp. 579–583). IOS Press.