Skin Synthesis: A Comparative Analysis of AI and Traditional Methods in Skin Type Classification

Gabe Gaw
Stanford University
Department of Computer Science
ggaw25@stanford.edu

Sherine Ismail
Stanford University
Department of Computer Science
sherinei@stanford.edu

Abstract

In this paper, we attempt to incorporate computer vision techniques into the dermatology domain, using various established Convolutional Neural Network models such as Residual Network (ResNet), Visual Geometry Group Network (VGGNet), and Mobile Network (MobileNet). The models are used to classify human skin types into three classes: oily, dry, and normal. We experiment with three datasets: a natural dataset in which images come from actual human skin examples, an AI-generated dataset in which images are generated using the text-to-image diffusion model Stable Diffusion v1-5, and a mixed dataset that contains both empirical data and AI-generated images. We employ transfer learning using the models to accomplish the skin-type classification task for the datasets. We conducted multiple experiments to enhance skin-type classification, compare the performance of various CNN models, and analyze their effectiveness on real versus AI-generated data. We found that variations of the ResNet model have higher accuracy rates across the empirical dataset, while variations of different models perform strongly on the AI-generated dataset. Additionally, we found that all three models tend to perform better on the natural dataset than the AI-generated dataset. We hope that the results from this paper will lead to further advancements in image classification tasks in dermatology and that experimenting with AI-generated data can be further studied in medical applications.

1. Introduction

Creating a personalized skincare routine can be challenging with the multitude of products available. Skincare products often target specific skin types, such as oily or dry, but many people are still unaware of their correct skin type. As a result, they use unsuitable products, potentially causing harm. A new study conducted by dermatology scientists at the Skin Trust Club found that nearly 2 in every 3 women do not know their correct skin type. We were inspired by this fact to experiment with how accurate computer vision models can be in classifying skin care types. More specifically, we aim to understand better if, and to what capacity, these models can outperform such individuals who incorrectly classify their skin type. As a result, an accurate classification system would facilitate accurate and accessible skincare recommendations, ultimately improving individuals' dermatological health and cosmetic outcomes.

Our project aims to develop a model that classifies skin types—oily, dry, or normal using skin images as input. We also address the issue of limited training data in current skin classification models by experimenting with AI-generated images. Specifically, we compare a fine-tuned model trained on real data with one trained on AI-generated data to assess the effectiveness of AI-generated images in skin type classification

We aim to accomplish three goals: improve existing skin type classification models using Deep Convolutional Networks, compare the performance of these models across various metrics (precision, recall, etc.), and experiment with AI-generated data for computer vision. To do so, we explore a variety of Deep Convolutional Networks such as ResNet50, VGGNet19, and MobileNet; through transfer learning, we are able to fine-tune these models on both the empirical dataset, the AI-generated dataset, and a mixed dataset, comparing model-wide performance. To further fine-tune these models, we conduct experiments that test various hyperparameters, architectures, and optimization strategies. For example, we compare the performance of ResNet50 against ResNet50 with dropout and against ResNet50 with batch normalization, dropout, and dense layers. We also compare optimization algorithms in models such as stochastic gradient descent with momentum versus Adam. Through these experiments, we are able to better understand the capabilities of such Deep Convolutional Networks in the dermatology domain.

2. Related Work

Numerous published works have explored the use of deep convolutional neural networks in the dermatology domain. However, they differ in their application within the space, and we found that the existing models for the skin type classification problem have many strengths and weaknesses that we could build on through our project.

2.1. Skin Type Classification Work

Firstly, Saiwaeo et al. developed their own skin type classification model using CNN deep learning algorithms, and they used transfer learning to test other existing CNN architectures. The model is trained on microscopic images of dry, normal, and oily skin, and while this allows the model to learn more complex features and nuances between the classes, it is also not consumer-friendly. Similarly, Saidah et al. also use a microscopic image dataset in order to develop their CNN model. The model consists of three hidden layers with fully connected layers and softmax activation, incorporating regularization techniques such as dropout. They used a total of 1600 images to classify data into three classes: oily, dry, and normal skin, and they achieved an accuracy rate of 99.5%. However, similar to Saiwaeo et al., the microscopic skin images are not accessible to everyday users, and the dataset the model is trained on is small in size. Lastly, Kothari et al. use CNNs to classify skin type with additional product recommendations. The developers used a tiny dataset of 80 skin images collected by web scraping and classified into oily and dry categories. These images were then processed to input into a face detector and facial feature extractor, and the resulting images (separate images of the forehead, left cheek, right cheek, and the nose) were inputted into the skin type classifer. Their developed process of breaking down the input into various images of facial features to feed into the CNN classifier is an incredible example of how the skin type classification problem can be further decomposed to optimize the model's ability to learn patterns based on smaller sections of the face.

2.2. Skin Disease Classification Work

There is also ample impressive work in the particular task of skin disease classification. For example, Sun et al. created a benchmark for automatic visual classification of clinical skin disease images; the authors used the largest dataset at the time (approximately 6600 images from 198 classes) and performed extensive analyses on the images using CNNs. They performed two experiments: (1) compare the influence of different baseline features through classification using hand-crafted features and (2) evaluate existing methods of fine-tuning models using transfer learning. Wu et al. further explores transfer learning for face skin disease classification by employing it on five mainstream CNN algorithms pretrained on ImageNet (ResNet-50, Inception-V3, DenseNet121, Xception, and Inception-ResNet-V2). The authors created a significantly larger dataset of approximately 2700 images assigned to six common skin diseases and used data augmentation to help support any data imbalances. They found that models pre-trained on other body part images are generally superior to models only trained on facial images. Ahmad et al. pioneer a new approach in the field, namely fine-tuning layers of ResNet152 and Inception-ResNet-v2 models with a triplet loss function. Not only do the authors have a large, diverse dataset of 12000 images with labels of 14 classes, but they also use techniques such as data augmentation, dropout, L2 regularization, and stochastic gradient descent with momentum to further optimize the models. Also, other work further applies these models by creating skin-type applications for users. For example, Velasco et al. build skin disease classification system on Android applications by applying transfer learning with MobileNet on 7 skin diseases. They use a significantly large dataset of about 3400 images across 7 classes, and they explore various sampling methods (undersampling versus oversampling) and preprocessing techniques on input data. However, the dataset is imbalanced and no measures were taken to correct the weights in the training process.

2.3. Other Skin Classification Work

In the dermatology domain, other work has been done to explore other skin classification tasks such as human skin detection, gender classification based on skin tone, and facial skin image classification. Salah et al. published a novel approach for human skin detection in which they trained a patch-based CNN that classifies the data as skin or non-skin, and then use a skin detection algorithm to scan the whole image and use a mask to extract and evaluate skin detection. For their input data, they created a new dataset from an existing, established dermatology dataset (SFA) and utilized diverse images with single and multiple objects. Additionally, Mustapha et al. improve the accuracy of gender classification based on skin tone using transfer learning on CNNs. They use a large dataset from FaceARG (approximately 6250 images) and divide the dataset into four classes: bright skin tone and female, bright skin tone and male, dark skin tone and female, and dark skin tone and male. They apply transfer learning and fine-tuning methods to the MobileNetV2 model, and the authors found that the dark experiment achieved the highest accuracy on the training dataset while the bright experiment scored the highest accuracy on the test set. While they discovered that the model performs significantly better with fine-tuning, there are extensive limitations to the lack of inclusivity in the classes. Lastly, Chin et al. explored facial skin image classification tasks using CNNs. The team used three varying CNN architectures: a two-layer CNN, a three-layer CNN, and transfer learning with LeNet-5. They classified the input data into three classes: good facial skin quality, bad facial skin quality, and face makeup. It was discovered that the three-layer CNN architecture has the highest recognition rate, but such model is trained on a relatively small dataset (approximately 300 images).

3. Dataset and Metrics

3.1. Datasets

In order to train our models via transfer learning, we searched for a publicly available dataset that contained accurate, realistic images of human skin across oily, normal, and dry types. We also wanted the dataset to be as diverse as possible with varying genders, races, and ages. While we were able to find a dataset through Kaggle, it is limited in its size. However, in the scope of the medical domain, access to relevant data is limited due to patient privacy regulations. This limitation prompted us to consider using AI-generated images as a complimentary dataset when comparing model performance. Note that all datasets use the same test set coming from the empirical dataset.

3.1.1 Empirical Dataset

As mentioned previously, we found a publicly available dataset of human skin on Kaggle called "Oily, Dry and Normal Skin Types Dataset" developed by Shakya Dissanayake. Originally, the dataset contained approximately 3150 images (Table 1). However, we decided to process the data further by removing ambiguous images. For example, if images contained skin care products, were blatantly edited, or contained non-human faces, we would remove them. After processing the dataset, we scraped the internet for free-to-use images for each class. We chose various images from microscopic skin, skin-care promotional photos, to pictures of friends. We added these images throughout the test, train, and validation sets to produce our current iteration of the dataset of about 2400 images (Table 2). As seen in Table 2, we have significantly less images for the dry class among the test, train, and validation sets. We resolve this data imbalance by applying weights to the classes throughout training. To further expand our dataset, we also experimented with incorporating AI-generated images to see its effects on our evaluation metrics.

Skin Type	Training Set	Validation Set	Test Set
Dry	652	71	35
Normal	1104	111	59
Oily	1000	80	40

Table 1. Unprocessed Kaggle Dataset Sizes by Skin Type and Set

Skin Type	Training Set	Validation Set	Test Set
Dry	280	35	32
Normal	723	100	96
Oily	893	102	109

Table 2. Processed Kaggle Dataset Sizes by Skin Type and Set

3.1.2 AI-Generated Dataset

In the medical field, one of the main challenges for computer vision models is the limited access to high-quality data. Privacy regulations and patient confidentiality often restrict access to accurately labeled data from professionals. While our current empirical dataset is larger than many existing datasets for the skin-type classification problem, we aimed to explore the use of AI-generated images as a potential solution to this widespread data limitation in medical computer vision. To do so, we set up a pipeline to generate images using Hugging Face's Stable Diffusion v1-5 Model. The Stable Diffusion v1-5 Model developed by Hugging Face is a text-to-image generative model that uses diffusion processes to iteratively refine a noisy image into a coherent image based on text-input prompts. The model takes as input a text prompt, the number of images to generate, the number of images per prompt, and a guidance scale. The guidance scale represents the degree of freedom the model has to generate images. The model is trained on a vast dataset of images and their corresponding captions, and it is excellent at capturing important details specified in the prompts. For each class (dry, normal, and oily), we experimented with various combinations of prompts and guidance scales in order to produce realistic images. One issue that we encountered was the instability of the model itself; many times it would generate images that are completely irrelevant to the prompt. As a result, we manually processed the dataset, discarding any images that were completely irrelevant, contained skin care products or non-human faces, as well as features that were too exaggerated/not realistic to human skin. For example, for many of the images in the dry class, we saw AI-generated images of skin that was excessively cracked, depicting an extreme and unnatural representation of dryness. We compare the distributions of the dataset before and after processing in Table 4 and 5 respectively.

Skin Type	Training Set	Validation Set
Dry	1000	160
Normal	1400	200
Oily	1300	160

Table 3. Unprocessed AI-Generated Dataset Sizes Before by Skin Type and Set

Skin Type	Training Set	Validation Set
Dry	617	96
Normal	738	77
Oily	641	64

Table 4. Processed AI-Generated Dataset Sizes Before by Skin Type and Set

3.1.3 Empirical and AI-Generated Dataset

An additional experiment we will conduct is applying transfer learning through a dataset that is mixed with empirical data and AI-generated images. Again, we hope to combat the ongoing roadblock of inaccessible medical data by extending existing datasets with computer vision techniques such as AI-image generation. The size of this dataset, approximately 4900 images, is significantly larger than our previous two datasets. The exact distribution is noted in Table 6.

Skin Type	Training Set	Validation Set
Dry	1018	63
Normal	1684	86
Oily	1745	88

Table 5. Empirical and AI-Generated Dataset Sizes Before by Skin Type and Set

3.2. Evaluation Metrics

Since we are implementing a classification task, our evaluation metrics will include precision, recall, and F1 scores. Such evaluation metrics will be reported across the varying models and datasets so that we will have a better understanding of the model's performance.

3.2.1 Precision

Precision is a measure of the accuracy of positive predictions made by the model. More specifically, it is the fraction of correctly predicted positive classifications to the total predicted positives. In the context of our project with the skin-type classification task, precision represents the proportion of correctly predicted instances among all instances that were predicted as particular skin type by the model.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- TP (True Positives) are the correctly predicted positive samples.
- FP (False Positives) are the incorrectly predicted positive samples.

A high precision score for a particular class suggests that the when the model predicts a certain skin type, it is usually correct. It will be important to record precision rates across classes to see if there are any biases in the training data.

3.2.2 Recall

Recall is the ratio of correctly predicted positive observations to all observations in the actual class. In the skin-type classification task, the recall metric measures how well the model identifies all the samples of a particular skin type.

$$\text{Recall} = \frac{TP}{TP + FN}$$

- TP (True Positives) are the correctly predicted positive samples.
- FN (False Negatives) are the actual positives that were not correctly identified by the model.

A high recall score for a particular class means that the model can identify most of the samples for that skin type.

3.2.3 F1 Score

F1 scores are the harmonic mean of precision and recall. We use it as a way to balance both precision and recall, especially since we have an uneven class distribution. The formula for calculating F1 scores are:

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

In this context, the F1 score represents a combined measure of the model's ability to identify each skin type accurately while also minimizing false positives.

4. Methods

4.1. Overall Approach

4.1.1 Image Preparation

a. For the real image dataset, we shuffled our modified Kaggle dataset containing labeled skin-type images and divided it into segments of 80% for training, 10% for validation, and 10% for testing.

b. For our AI image dataset, we created a training and a validation set from our generated images. Since we were using our test set from our real dataset, we matched the size of the AI validation set to the real test set, leaving the rest of the generated images to make up the training set.

c. For our real and AI combined dataset, we kept our test set from our real dataset. To create the train and validation sets, we shuffled our real and AI datasets together, removed any images that overlapped with those in our test set, and then created a validation set to match the size of our test set. The rest of the images were then put into the training set.

4.1.2 Image Pre-Processing

For all three datasets, we followed the same image preprocessing approach, which included image resizing, normalization, and random flips during training.

4.1.3 Evaluation Metrics

In addition to evaluating the classification accuracy on the validation set, we evaluated class-wise accuracy, class-wise precision, class-wise recall, and class-wise F1 score. These metrics help us measure the performance of the model on each class, giving insights into its ability to identify positive instances for each class and make correct positive predictions.

4.1.4 Model Building

Loss Function We decided on cross-entropy loss because it is widely recognized to be well-suited for neural network classification problems. It effectively handles probabilistic outputs and is sensitive to misclassification. Cross-entropy loss quantifies the difference between the predicted probability distribution and the true distribution, helping our model learn to output probabilities that closely match the actual distribution of the data. For our skin type classification problem, our loss function is the following:

$$L = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c}) \tag{1}$$

To account for class imbalances, we calculated class weights from the training sets to give more importance to underrepresented classes during training. The weight for each class is calculated as the inverse of its frequency:

$$w_i = \frac{N}{n_i} \tag{2}$$

where N is the total number of samples in the training set. These weights are then normalized so that their sum equals 1:

$$\hat{w_i} = \frac{w_i}{\sum_j w_j} \tag{3}$$

For the loss function, using cross-entropy loss as an example, the implementation of class weights would modify the original function from (4) to (5):

$$L = -\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c}\log(p_{i,c}) \tag{4}$$

$$L = -\sum_{i=1}^{N}\sum_{c=1}^{C} w_c \cdot y_{i,c}\log(p_{i,c}) \tag{5}$$

effectively adjusting the loss contribution from each class.

4.2. Transfer Learning

The real image dataset was used to test different transfer learning models with respective modifications to determine the best one at classifying skin types. The three different pre-trained models we used for transfer learning were ResNet50, VGG19, and MobileNet. We followed a pytorch tutorial, and modified functions we wrote in class problem sets to set up transfer learning. For all of the different model architectures we tested, we used the versions pre-trained on ImageNet as a starting point, hoping to leverage well-learned features for our skin type classification task.

4.2.1 ResNet50

We chose ResNet50 as one of our models to test since it is widely used for image classification and considered one of the most effective architectures.
ResNet 50 is a deep convolutional network (CNN). The 50 layers of ResNet50 comprise convolutional layers, batch normalization layers, ReLU activations layers, and fully connected layers. In addition to those layers, ResNet50 employs residual blocks to solve the common vanishing gradients problem in deep networks. In residual learning, the network learns a residual function instead of directly trying to learn underlying mappings. The residual blocks that facilitate this contain a few convolutional layers where the input to each layer is added to the output of the block, forming a skip connection that helps with gradient backpropagation. To modify the network for our skin type classification problem, we modify the final fully connected layer.

4.2.2 VGG19

We chose VGG19 to test because it is a relatively simple model that is less computationally expensive, and we wanted to see how it would perform against a deeper model like ResNet50.
VGG19 is a deep convolutional neural network (CNN). Compared to ResNet50, VGG19 is much more simple with only 19 layers broken down into 16 convolutional layers and 3 fully connected layers. Furthermore, VGG19 uses small 3x3 convolutional filters throughout the network. VGG19 is characterized by its uniform design where repeating blocks of 3x3 convolutional layers stacked on top of each other followed by a max-pooling layer make up the network. The network ends with 3 fully connected layers, and we modify the last fully connected layer to match our skin type classification problem.

4.2.3 MobileNet

We chose MobileNet to test because it is specifically designed for resource-constrained environments, and given our resource constraints, we wanted to see how it would stack up against the other networks.
MobileNet is a class of efficient deep neural networks designed for mobile and embedded vision applications. A key feature of MobileNet is its use of depthwise separable convolutions, which are standard convolutions factorized into a depthwise convolution and a pointwise convolution. This feature reduces the number of parameters and computational cost. Similar to the other 2 networks, MobileNet ends with fully connected layers, and we modify the last one to suit our skin type classification problem.

For each of the three different models, we implemented 2 additional modifications to create a total of 9 models. The first modification was to add a dropout layer to help prevent overfitting. Dropout involves randomly setting a fraction of the input units to zero during each training cycle to force the model to learn more robust features. We decided to implement this modification since none of the models above have dropout layers in their architecture. The second modification was to add a series of dense layers consisting of batch normalization, ReLU, and dropout layers on top of the existing architecture to facilitate fine-tuning for out specific problem.

4.3. Optimizer Testing

We first evaluated two optimization algorithms we found the most effective from our coursework, stochastic gradient descent and Adam.

4.3.1 Stochastic Gradient Descent

SGD is a variant of traditional gradient descent, and is effective for large-scale problems. SGD updates parameters based on the gradient of the loss computed from randomly sampled batches. The update rule for SGD is the following: $\theta = \theta - \eta \nabla_\theta L(\theta)$. With SGD, we also implemented modifications such as momentum and learning rate scheduling to improve the performance and convergence of the model.

4.3.2 Momentum

Momentum adds a fraction of the previous update to the current update to smoothen oscillations and prevent SGD from getting stuck at local minimums. To incorporate momentum into SGD we calculate $v_t = \beta v_{t-1} + (1-\beta)\nabla_\theta L(\theta)$, where β is the momentum factor, and the update rule becomes $\theta = \theta - \eta v_t$. We tested multiple values of momentum factors during training.

4.3.3 Learning Rate Scheduling

Learning rate scheduling allows us to adjust the learning rate during training to improve convergence, we tested out different step sizes and gammas where the learning rate would decrease by a factor of gamma for every step size.

4.3.4 Adam

Adam (Adaptive Moment Estimation) is a further extension of SGD that computes adaptive learning rates for each parameter. It essentially maintains two moving averages to adaptively adjust the learning rate for each parameter.
The first moment $m_t = \beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta L(\theta)$ represents the exponential moving average of the gradients, and the second moment $v_t = \beta_2 v_{t-1} + (1-\beta_2)(\nabla_\theta L(\theta))^2$ represents the exponential moving average of the squared gradients. Before performing the updates we then performing bias correction, $\hat{m}_t = \frac{m_t}{1-\beta_1^t}$ and $\hat{v}_t = \frac{v_t}{1-\beta_2^t}$. Then finally

the update rule is $\theta = \theta - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t}+\epsilon}$.
For the two optimizers, we tested them out on all 9 models on our real image dataset to find the best model before fine-tuning.

4.4. Fine-tuning

For fine-tuning we tested different values for our hyperparameters on the real image dataset. The parameters we toggled included the learning rate, regularization constant, dropout keep ratio, learning rate decay factor (gamma), learning rate decay step size, and the number of epochs.

4.5. Model Evaluation and Visualization

With our best model and best hyperparameters, we trained three separate models on our three datasets, and extracted test accuracy on unseen data consisting of real images. Furthermore, we generated saliency maps to visualize our models' learned features.

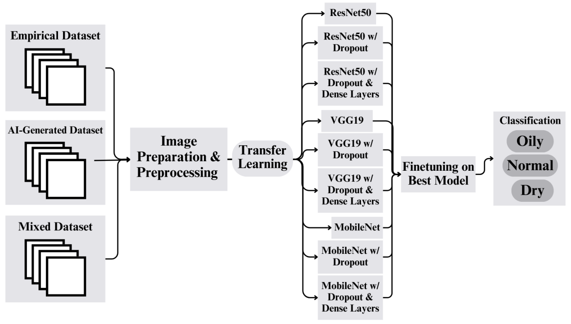

Figure 1. Our overall approach to the skin-type classification task.

5. Results

5.1. Model Architecture

5.1.1 Testing Optimizers

As explained in the methods section, we set up 9 different transfer learning models, 3 variations of each pre-trained model from ResNet50, VGG19, and MobileNet. In order to determine whether Adam optimizer or SGD optimizer was more suitable for our skin type classification task, we tested the performance of both optimizers using a learning rate of 0.1, and a weight decay of $1e^{-4}$ which we found to be good starting points. The experiments were performed on our real image dataset. The results are displayed in figures 2 and 3.

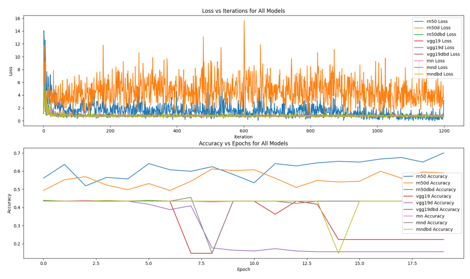

Figure 2. Model performance using Adam Optimizer.

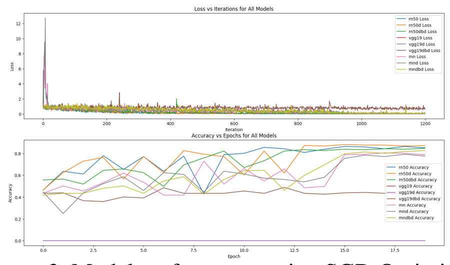

Figure 3. Model performance using SGD Optimizer.

As seen from the graphs, the SGD optimizer performed better on most models, achieving an accuracy of over 80% for its best-performing model while Adam only achieved an accuracy of around 70% for its best-performing model. As a result, we decided to use SGD optimizer for our following experiments.

5.1.2 Testing pre-trained models

With SGD as our optimizer, the top 3 performing models were all variations of the pre-trained ResNet50 model. Their respective best validations accuracies along with the other models following 20 epochs of training are displayed in table 7.

Model variation	**Val Accuracy (%)**
ResNet50 (rn50)	86.5
ResNet50 + dropout (rn50d)	88.19
ResNet50 + dense layers (rn50dbd)	84.81

Table 6. Model variations performance on validation set.

We then proceed with hyper-parameter tuning on our best model which is ResNet50 + dropout. These experiments were conducted on the real image dataset.

5.1.3 Hyper-parameter fine-tuning

To obtain a higher validation accuracy for our ResNet50 + dropout model, we added a learning rate scheduler, regularization, and momentum updates to SGD. As our parameters for fine-tuning, we tested out different learning rates, regularization constants, dropout keep ratios, learning rate decay factors, learning rate decay step sizes, and epochs. Due to computational constraints, we could only test a limited combination for these parameters. Our results are in table 7.

5.1.4 Model Training on all Datasets

Using our ResNet50 + dropout model and the hyperparameters from testing, we trained on our 2 additional datasets, our AI images dataset, and our AI + real dataset. The results are shown in table 8.

Train Dataset	**Val Accuracy (%)**	**Test Accuracy (%)**
Real	91.56	89.45
AI	81.52	30.80
Mixed	61.60	81.01

Table 8. Models trained on different datasets performance on the validation set and test set.

Iteration	Epochs	LR	LR decay factor	LR step size	Regularization constant	Val Accuracy (%)	Class-wise F1 score [dry, normal, oily]
1	13	0.00012	0.04267	1	0.34266	0.89451	[0.7879, 0.8780, 0.9261]
2	11	0.00001	0.01714	4	0.00626	0.8861	[0.8000, 0.8718, 0.8879]
3	17	0.00026	0.00264	6	0.0991	0.89451	[0.7692, 0.8687, 0.9005]
4	15	0.00009	0.00357	7	0.282	0.9156	[0.8696, 0.9261, 0.9208]
5	12	0.00005	0.01706	3	0.55391	0.8819	[0.7879, 0.8780, 0.9261]

Table 7. Iterations of hyperparameter fine-tuning.

5.2. Qualitative Results

In addition to classification accuracy, we generated predictions and saliency maps to visualize our results. Saliency maps visualize the regions of an input image that our model considers most influential for making predictions. In the following sections, we will provide examples of saliency maps per class and per dataset to understand potential differences in the model's learned patterns/features.

5.2.1 Saliency Map for Model Trained on Real Dataset

In the saliency map of the image with dry skin, the model highlights flakey and dry skin patches, and when overlayed with the original image, we see that the model correctly highlights the relevant portions of the image (Figure 4). For the normal image, the saliency map highlights on the man's entire face which makes sense as there are no relevant dry or oily patches in the image (Figure 5). The saliency map of the image with oily skin correctly emphasizes the oily parts of the woman's face, namely the chin, around the nose/cheeks, and the forehead (Figure 6).

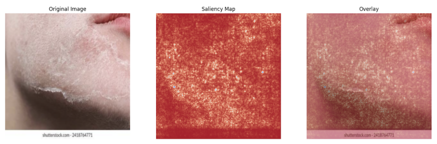

Figure 4. Saliency map on dry image.

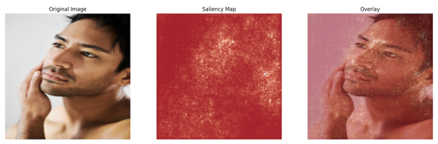

Figure 5. Saliency map on normal image.

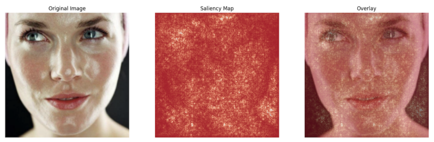

Figure 6. Saliency map on oily image.

5.2.2 Saliency Map for Model Trained on AI Dataset

We generated new saliency maps for the same images as before, but now for the model that was trained on the AI dataset alone. In the image of the dry skin, the saliency map does not highlight a significant portion of the dry patches, supporting our belief that the AI-trained model does not generalize well to real images (Figure 7). The saliency map for the image of normal skin focuses on the man's face but also some of the background (Figure 8). The model also correctly highlights the oily patches of the skin in the oily image (Figure 9).

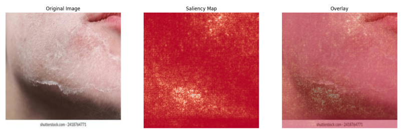

Figure 7. Saliency map on dry image.

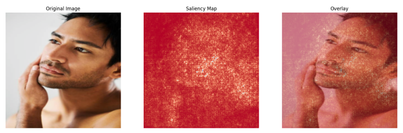

Figure 8. Saliency map on normal image.

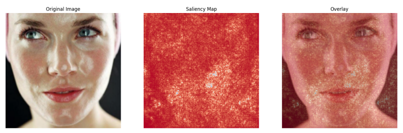

Figure 9. Saliency map on oily image.

5.2.3 Saliency Map for Model Trained on Mixed Dataset

Lastly, we provide examples of saliency maps for the model trained on the mixed dataset. Across all the images, we found that the highlighted areas are more concentrated than those of the previous saliency maps. For example, in the oily image, we see that the model solely highlights oily patches whereas the other models would capture less concentrated patches across the entire image (Figure 12). This is also prevalent in the normal image as well, the model only significantly highlights a few portions of the face (Figure 11). However, in the dry image, the model fails to identify key dry patches in the image (Figure 10).

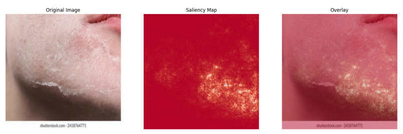

Figure 10. Saliency map on dry image.

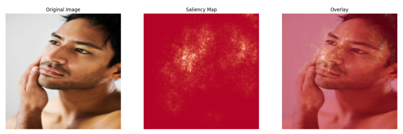

Figure 11. Saliency map on normal image.

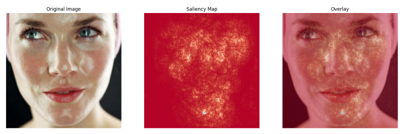

Figure 12. Saliency map on oily image.

5.2.4 Predictions of Model Trained on Real Dataset

We have provided examples of images in which the model trained on the real dataset correctly classifies the class.

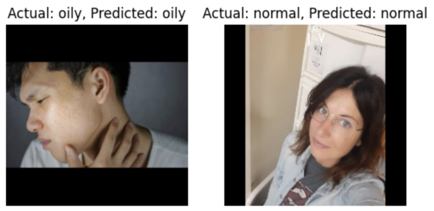
Actual: oily, Predicted: oily
Actual: normal, Predicted: normal

Figure 13. Model predictions.

5.2.5 Predictions of Model Trained on AI Dataset

In the following figure, we showcase instances where the model trained on the AI dataset accurately identifies and misclassifies skin types.

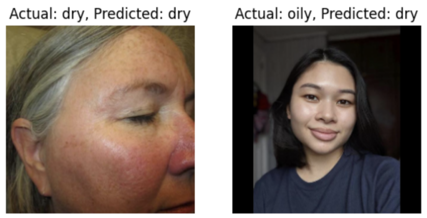

Figure 14. Model predictions.

5.2.6 Predictions of Model Trained on Mixed Dataset

Lastly, for the model trained on both real and AI-generated images, we provide two examples where the model correctly identifies the skin type.

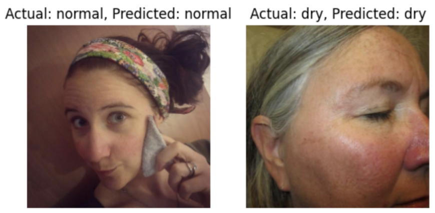
Actual: normal, Predicted: normal
Actual: dry, Predicted: dry

Figure 15. Model predictions.

6. Conclusion and Future Works

In this project, we evaluated the performance of various models on a skin-type classification task using different datasets, deep learning techniques, and hyperparameters. We used three datasets including real images, AI-generated images, and a mixed dataset containing both types of photos. To reiterate, we aimed to (1) improve existing skin type classification models using CNNs, (2) compare the performance of these models across varying metrics (precision, recall, etc.), and (3) experiment with AI-generated data. In terms of improving existing skin type classification models, after experimenting with and fine-tuning different models such as ResNet50, VGG19, and MobileNet, we were able to obtain a validation accuracy of 91.56% and a testing accuracy of 89.45% with ResNet50 (Table 7, Table 8). More specifically, we applied regularization techniques like incorporating dropout layers and fine-tuned the model's hyperparameters to achieve such accuracy rates. Experimenting with AI-generated data allowed us to better understand the capacity to which these images can augment, or even replace, real data in the training stages. Models trained solely on AI-generated images struggled to generalize to real-world data, reflecting that AI images are not yet a viable substitute for real images in the dermatology domain. However, we saw improvements with the mixed dataset; while the mixed model did not perform as well as the model trained on real images, it still demonstrated reasonably high test accuracy of about 81.01%. This suggests that AI-generated images could potentially be used to supplement training data, creating a more robust model by introducing variability akin to noise. Despite attempts to expand our current iteration of the datasets, we believe that the size of these datasets needs to be larger for future work. Expanding the datasets with more diverse and representative images could further enhance the models' performance and generalization. Additionally, for future work, we would like to further explore the ability of AI-generated images to supplement training data; future work could focus on improving the realism of AI-generated images or exploring the optimal ratio of real to AI-generated images to maximize the model's performance. Overall, our study significantly improved our understanding of not only sophisticated CNNs, but also the potential to use AI-generated images in a domain where data is sometimes inaccessible.

7. Contributions Acknowledgements

Both authors Sherine Ismail and Gabe Gaw worked on the project evenly. Together, we developed the experimental design, methods, and code for the experiments. Gabe Gaw ran the experiments and generated the quantitative results. Sherine Ismail created the qualitative results including the saliency maps and prediction images. Sherine Ismail wrote the abstract, introduction, and datasets/metrics sections. Gabe Gaw wrote the methods, results, and conclusion/future works. We worked together to find previous work and write the related works section.

References

[1] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer, "High-Resolution Image Synthesis With Latent Diffusion Models", *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 10684-10695.

[2] Sirawit Saiwaeo, Sujitra Arwatchananukul, Lapatrada Mungmai, Weeraya Preedalikit, Nattapol Aunsri, "Human skin type classification using image processing and deep learning approaches", *Heliyon*, vol. 9, no. 11, 2023, article e21176, ISSN 2405-8440, https://doi.org/10.1016/j.heliyon.2023.e21176.

[3] Shakya Dissanayake, "Oily, Dry and Normal Skin Types Dataset", https://www.kaggle.

com/datasets/shakyadissanayake/oily-dry-and-normal-skin-types-dataset, Accessed 17 May 2024.

[4] Saidah, Sofia, et al. "Facial Skin Type Classification Based on Microscopic Images Using Convolutional Neural Network (CNN)." Proceedings of the 1st International Conference on Electronics, Biomedical Engineering, and Health Informatics, edited by Triwiyanto et al., Springer, 2021, pp. 75–83. Springer Link, https://doi.org/10.1007/978-981-33-6926-9_7.

[5] Kothari, Arya, et al. "Cosmetic Skin Type Classification Using CNN With Product Recommendation." 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), 2021, pp. 1–6. IEEE Xplore, https://doi.org/10.1109/ICCCNT51525.2021.9580174.

[6] Sun, Xiaoxiao, et al. "A Benchmark for Automatic Visual Classification of Clinical Skin Disease Images." Computer Vision – ECCV 2016, edited by Bastian Leibe et al., Springer International Publishing, 2016, pp. 206–22. Springer Link, https://doi.org/10.1007/978-3-319-46466-4_13.

[7] Wu, Zhe, et al. "Studies on Different CNN Algorithms for Face Skin Disease Classification Based on Clinical Images." IEEE Access, vol. 7, 2019, pp. 66505–11. IEEE Xplore, https://doi.org/10.1109/ACCESS.2019.2918221.

[8] Ahmad, Belal, et al. "Discriminative Feature Learning for Skin Disease Classification Using Deep Convolutional Neural Network." IEEE Access, vol. 8, 2020, pp. 39025–33. IEEE Xplore, https://doi.org/10.1109/ACCESS.2020.2975198.

[9] Velasco, Jessica, et al. "A Smartphone-Based Skin Disease Classification Using MobileNet CNN." International Journal of Advanced Trends in Computer Science and Engineering, Oct. 2019, pp. 2632–37. arXiv.org, https://doi.org/10.30534/ijatcse/2019/116852019.

[10] Ben Salah, Khawla, et al. "A Novel Approach for Human Skin Detection Using Convolutional Neural Network." The Visual Computer, vol. 38, no. 5, May 2022, pp. 1833–43. Springer Link, https://doi.org/10.1007/s00371-021-02108-3.

[11] Mustapha, Muhammad Firdaus, et al. "Improving the Accuracy of Gender Classification Based on Skin Tone Using Convolutional Neural Network: Transfer Learning (CNN-TL)." Proceedings of the 12th International Conference on Robotics, Vision, Signal Processing and Power Applications, edited by Nur Syazreen Ahmad et al., Springer Nature, 2024, pp. 493–98. Springer Link, https://doi.org/10.1007/978-981-99-9005-4_62.

[12] Chin, Chiun-Li, et al. "Facial Skin Image Classification System Using Convolutional Neural Networks Deep Learning Algorithm." 2018 9th International Conference on Awareness Science and Technology (iCAST), 2018, pp. 51–55. IEEE Xplore, https://doi.org/10.1109/ICAwST.2018.8517246.