SKIN CANCER DETECTION USING DEEP LEARNING

Paarth Kashyap
Student# 1009034670
p.kashyap@mail.utoronto.ca

Qi Zheng
Student# 1008329667
qqi.zheng@mail.utoronto.ca

Kuval Brar
Student# 1009102659
kuval.brar@mail.utoronto.ca

Akarsh Singhal
Student# 1008035154
akarsh.singhal@mail.utoronto.ca

ABSTRACT

This document will propose Group 38's APS360 Final Project. The project aims to develop a deep-learning model that will classify different types of skin lesions while differentiating the cancerous ones. The document will highlight data processing, the architecture of the artificial neural network suggested, baseline model, results, and all other appropriate sections from the "Project Final Report Handout and Rubric".

—-Total Pages: 10

1 INTRODUCTION

Skin cancer is one of the most common types of cancer, and early detection is crucial for effective treatment. Typically, doctors rely on visual inspections and biopsies for diagnosis, which can be both time-consuming and prone to human error. While the traditional diagnostic process for skin cancer is effective, there is considerable potential to enhance it further. By incorporating advanced technologies, we can significantly improve the accuracy and efficiency of diagnosing skin cancer. These innovations would support doctors in making more informed and timely decisions, ultimately benefiting patient outcomes.

1.1 GOAL

Team 38's goal with the project is to create an artificial neural network that can classify skin lesions as well as label them as either benign or malignant using dermoscopic images. The team aims to achieve a fully trained artificial neural network that can successfully identify the types of skin cancer a patient may have and communicate results to patients and/or dermatologists enabling them to take immediate action if needed. Dermatologists will be able to utilize this alternative automatic procedure for skin cancer diagnosis to make their work more efficient and reliable as well as deliver the patient a less stressful and quicker response for their diagnosis.

1.2 IMPORTANCE

This project is important to the team and is motivated to develop it for several reasons:

- Effective and Early Response: Convolutional Neural Networks (CNNs) can analyze skin lesions with high accuracy, avoiding late diagnoses while enhancing diagnostic efficiency. This project aims to save lives and reduce misdiagnosis.
- Increased Accessibility and Affordability: The project can be integrated into mobile applications making it accessible to people in areas without full medical facilities. The automatic diagnosis also keeps the cost low for patients as fewer visits to specialists and dermatologists will be necessary.

- Support for Medical Professionals: This project can assist dermatologists and skin specialists in diagnosing patients by providing a reliable second opinion, a quick response, and reducing the risk of human error.

1.3 Rationale

Deep learning is a rational approach to help detect skin cancer since CNNs are exceptional in image recognition and can classify things effectively from a large dataset. Skin specialists and dermatologists spend years learning and practicing medicine to help correctly diagnose patients, while CNNs can do the same diagnosis in much less time with equal if not more accuracy. Also, our project will continue to improve as more data gets collected, the model's real-time diagnostic abilities are improved, providing better support for medical professionals, ultimately enhancing patient care.

2 Illustration

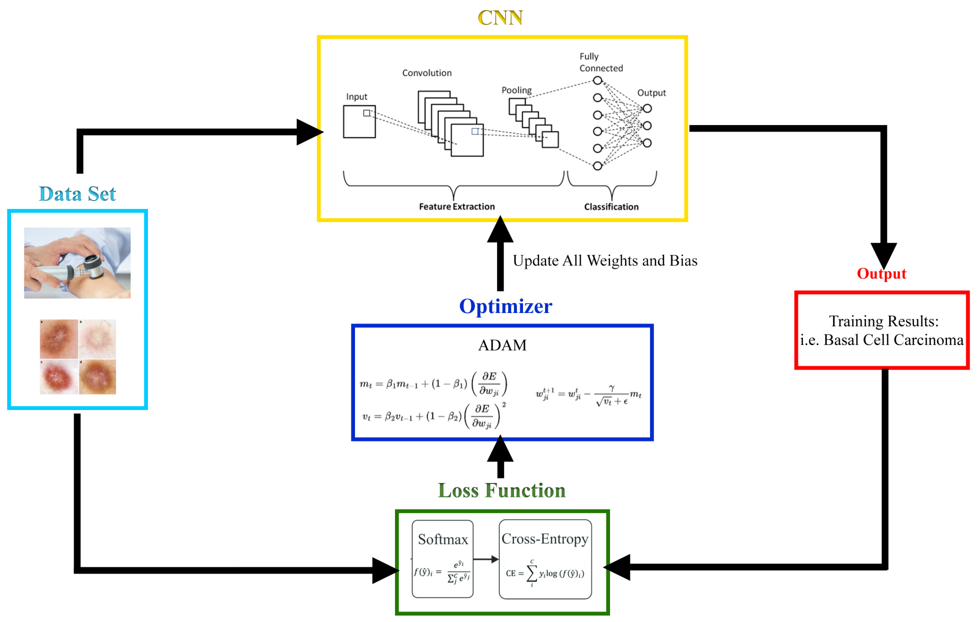


Figure 1: Summary of Model Functionality

3 Background & Related Work

Skin cancer is becoming a significant health issue, and early detection is crucial for successful treatment. In recent years, many studies have used machine learning techniques like Support Vector Machines (SVM) and Convolutional Neural Networks (CNN) to classify skin cancer. This section reviews key research that has influenced our project's development.

3.1 Skin Cancer Classification Framework Using Enhanced Super Resolution Generative Adversarial Network and Custom Convolutional Neural Network

This paper performed skin cancer classification using a custom Convolutional Neural Network (CNN) utilizing deep learning-based algorithms. Images were initially collected from HAM10000 dataset to be resized and pre-processed for better image resolution and have the neural networks perform on them. The dataset consisted of 7 different diagnosis classes; (Actinic Keratosi, Basal Cell Carcinoma, Benign Keratosis-Like Lesions, Dermatofibroma, Melanocytic Nevi, Vascular Lesions, and Melanoma). Using a train test ratio of 80:20 which averaged out to the final accuracy tests of 95% and the highest CNN algorithm with 98% accuracy was ESRGAN-based CNN (1).

3.2 Multi-class Skin Cancer Classification Architecture Based on Deep Convolutional Neural Network

The study here focuses on using CNN and SVM models, including InceptionV3, Xception, DenseNet, MobileNet, ResNet-50, and VGG-16, for classifying skin lesions into seven different types. The researchers used the HAM10000 dataset, which contains over 10,000 images, providing 9617 samples for training. The results showed InceptionV3 achieving the highest accuracy at 90%, while VGG-16 had the lowest at 76%, with an average accuracy of 84% across all models (2).

3.3 Machine Learning Approaches for Skin Cancer Classification from Dermoscopic Images: A Systematic Review

This study reviewed approximately 68 articles related to deep learning models that work on detecting skin cancer with the majority focus on CNN models. Some of the key findings from this study were that CNN models outperformed in being able to classify skin cancer types from dermoscopic images. Models like Inception, and ResNet performed well but with pre-trained data and refined that which also helped reduce training time. The dataset used was the HAM10000 (3).

3.4 An Effective Skin Cancer Classification Mechanism via Medical Vision Transformer

Using a two-fold approach, this study first employed CNN models (VGG, AlexNet) and then SVM with an RBF kernel. Initially, the model was deployed as a web and mobile application, gathering image samples for effective training. The Medical Vision Transformer (MVT) incorporated both CNN and SVM techniques to classify the images into the correct cancer categories. The CNN models achieved an accuracy of 91%, while the SVM classifiers reached 86.6% after 100 epochs. This method allows greater accessibility by enabling users to upload their images for skin cancer classification through online applications (4).

3.5 Detection of melanoma in dermoscopic images by integrating features extracted using handcrafted and deep learning models

Melanoma skin cancer is the key focus of this study which is detected using CNN classifying. Similar to other related works, this has methods to remove hair from the dermoscopic images in the pre-processing phase. This study performed 94.9% accuracy on the HAM10000 and 98% on the PH2 dataset. The Ph2 dataset, only focuses on melanoma, having 40 melanoma images and 160 non-melanoma images. ResNet50V2 and EfficientNet-B0 were used for the classifications as well as pre-processing (5).

4 Data Processing

The data source used is called "Skin Cancer MNIST: HAM10000". This dataset was chosen to be our source for data as it contains dermatoscopic images of common pigmented skin lesions. It consists of 7 different diagnosis classes; Actinic Keratosis, Basal Cell Carcinoma, Benign Keratosis-Like Lesions, Dermatofibroma, Melanocytic Nevi, Vascular Lesions, and Melanoma. The source of this dataset is originally Harvard Dataverse and it was authored by researcher Philipp Tschandl (6). The following are the steps that were taken to bring the data to the project (7):

1. Go to the Kaggle account and create a new API token, it will automatically download a .json file.
2. Create a Google Colab project file. Type the commands:

```
! pip install -q kaggle
from google.colab import files
files.upload()
```

3. Run the commands from step 2. Upon running you will be requested to upload a file, so upload the .json file downloaded from step 1.
4. Make a directory named kaggle and copy kaggle.json file there by typing the following commands and run:

```
! mkdir ~/.kaggle
! cp kaggle.json ~/.kaggle/
```

5. Run the following command to change the permissions of the file:

```
! chmod 600 ~/.kaggle/kaggle.json
```

6. To download the data, type the following and run:

```
! kaggle competitions download -c 'name-of-dataset'
```

7. Now unzip the data and create a directory for it by typing the following:

```
! mkdir dataset
! unzip name-of-dataset.zip -d dataset
```

After successfully importing the dataset, the images are loaded into a custom dataset class from the master directory, which holds the location paths, tensors, and labels for all images. During initialization, each image is resized to 224x224 pixels, normalized, and stored as PyTorch tensors, while the truth labels are converted to one-hot encodings.

Since the original HAM10000 dataset is not balanced or stratified, duplication and data augmentation are necessary to create a quality stratified dataset for the model. To ensure that the testing data remains unseen by the model, 20% of the images are extracted into a subset for later testing, leaving 80% for augmentation.

Also, images of less frequent classes were augmented to produce more samples. The reason for this is to balance the dataset so that there is a similar number of samples in each class.

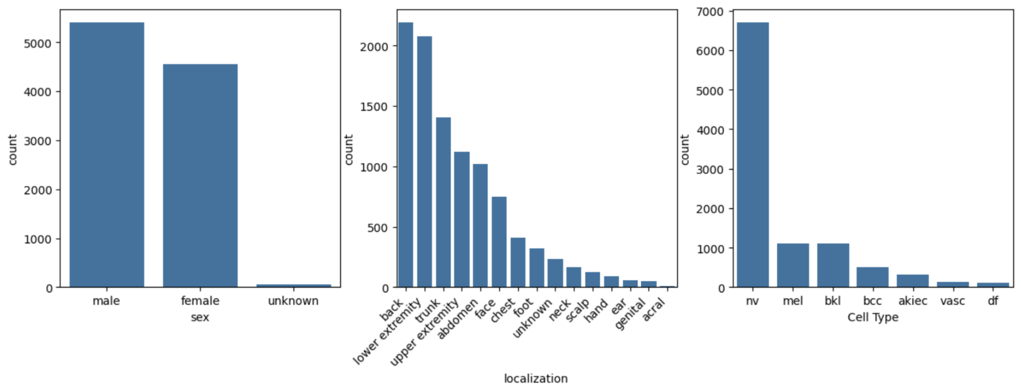


Figure 2: Visualizing the Dataset Info

Using the albumentations library, various data augmentation techniques are applied, including random 90-degree rotations, brightness and contrast adjustments, blur variations (such as motion blur and medium blur), and Gaussian noise filters. These augmentations are only applied to duplicate images from minority classes. It should be noted that the augmentations applied do not alter the image size, only the values of individual pixels as image size must stay consistent to be fed to the model. After augmentation, the images are normalized and converted back to tensors, and all augmented data is collected into a single dataset.

The augmented dataset is then split into training and validation sets using a stratified 60-20 split, with the initial 20% of the original dataset reserved for testing. This 60-20-20 split ensures sufficient data for training to learn patterns and features, as well as ample data for validation and testing. The testing data is kept entirely unseen and unaugmented to maintain the integrity of the model's evaluation.

To better understand the dataset, visualizations are performed on both the untouched and augmented data. We use Matplotlib, Pandas, Glob, and Seaborn to graph and plot the data. Figure 3 visualizes the distribution of sex, localization, and class imbalance among skin lesions. The seven unique skin lesion classes are mapped to numeric values for consistency throughout the notebook.

After applying the duplication and augmentation, the initially imbalanced data (NV is significantly more present than DF class type) is now balanced and ready to be loaded. Using a 60-20-20 stratified split, data is loaded into data loaders and fed into the training loop to train the model. it should be noted that during this process, the images are converted into NumPy arrays, torches, and tensors to comply with the different requirements of the imported libraries that split and load the data loaders. The dataset was split into training, validation, and testing sets, with the testing set containing data that the model had never seen before. Thus, we used the testing class data to test the model's performance.

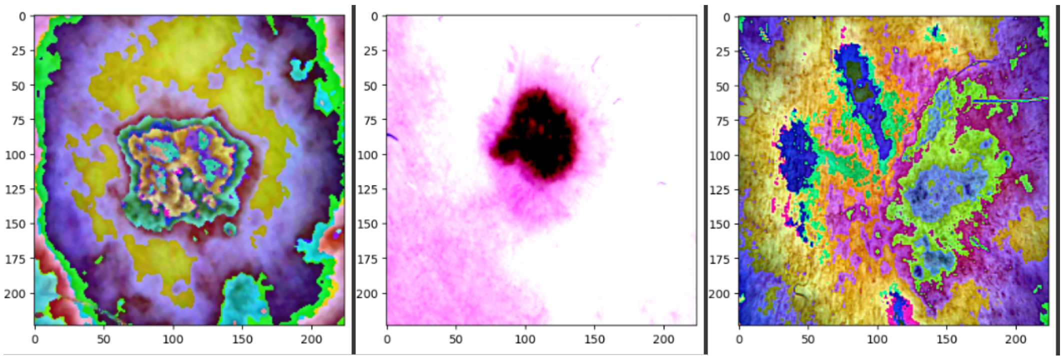


Figure 3: Augmented Images

5 Architecture

5.1 Description

The type of neural network that is most suitable for our skin cancer multi-classification project is a Convolutional Neural Network (CNN). This is the most efficient artificial neural network for detecting features in images, and in this case, detecting different types of skin lesions in dermoscoptic images. A CNN does this by utilizing a kernel and using it to scan through an image at a specific rate depending on the stride. Eventually, the CNN model will learn features and generate a feature map, ending the encoding phase. This feature map can then be flattened and put through a fully connected neural network for the classification phase of the model. This is where the model will classify the different types of skin lesions.

5.2 Relevant Components

The optimizer that will be used is Adaptive Moment Estimation (ADAM) as this helps rapidly converge to a solution, in other words, converging to the optimal weights and bias quicker. The regularization method used will be dropout. This prevents the model from over-fitting and memorizing the training data. As this will be a multi-classification problem, the activation functions that will be used are ReLU and softmax for the last layer. In addition, the loss function that is most suited for this multi-classification project will be Cross-Entropy (CE). Hyperparameters specific to CNN that need to be considered are kernel size, number of kernels, strides, padding, etc. Other hyperparameters to be considered when in the learning phase are batch size, number of epochs, learning rate, etc. A specific number of iterations of the model can be done using randomly selected hyperparameters. The set of hyperparameters from the model that performed the best will be used.

5.3 Specific Details

- Input Layers: 224 x 224 pixels resolution RGB images.
- Convolutional Layers (3): Multiple layers that each use ReLU activation function to introduce non-linearity and mitigate the vanishing gradient problem.
- Batch Normalization Layers (3): Increase the model's generalization capability.
- Pooling Layer: Reduction of spatial dimension by half after max pooling layer is applied. 2x2 window, stride.
- Dropout Layers (2): To prevent overfitting.
- Flattening Layer: 2D feature maps to 1D feature vector for classification purposes.
- Fully Connected Layers (3): Uses ReLU activation function to mitigate vanishing gradient problem.
- Output Layer: Used a Softmax layer to produce normalized probabilities for multiple classes (7 classes).
- Optimizer: Adam optimizer for efficient and stable convergence.
- Loss Function: Cross-Entropy Loss for multi-class classification.
- Batch Size: Used 128 to have good speed and performance at the same time.
- Epochs and Learning Rate: 20 and 1e-3

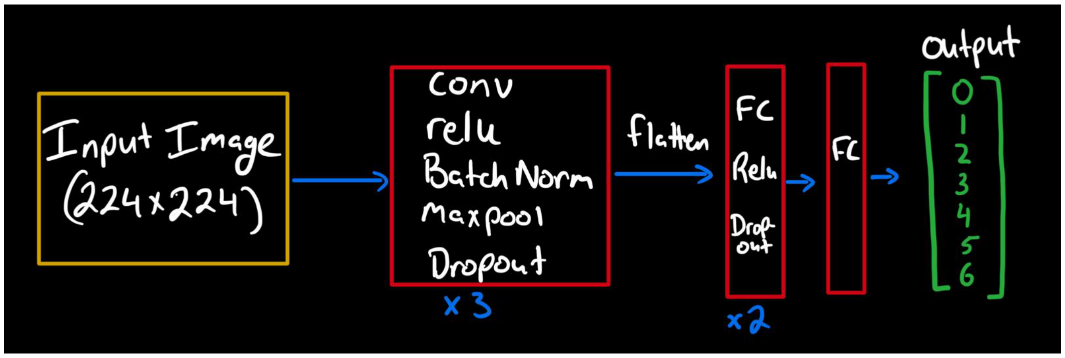
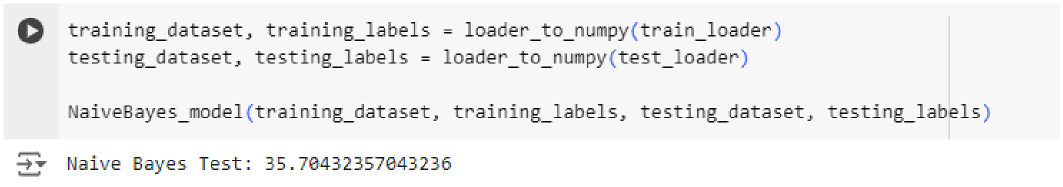


Figure 4: Architecure of CNN Model Used

6 Baseline Model

The team chose Gaussian Naive Bayes (GNB) for its ability to classify multiple types of skin cancer, though it was expected to perform poorly, achieving only 35.7% accuracy. To establish a stronger benchmark, we also used ResNet50 with pretrained weights as a baseline. Since ResNet50 was initially trained on a wide range of images, we made slight modifications to suit our needs. The final dense layer was adjusted to produce 7 outputs, and the model was trained for 3 epochs with a 1e-3 learning rate and a batch size of 128, resulting in 74.2% accuracy. This minimal training allowed us to establish valid baseline results for skin lesion classification.



```
training_dataset, training_labels = loader_to_numpy(train_loader)
testing_dataset, testing_labels = loader_to_numpy(test_loader)

NaiveBayes_model(training_dataset, training_labels, testing_dataset, testing_labels)
```

```
Naive Bayes Test: 35.70432357043236
```

Figure 5: Naive Bayes Test Score

7 Quantitative Results

The team created 4 different models with different values for parameters and different numbers of layers in each to figure out which one achieved the best performance metrics. After rigorous training, the team settled with SkinCancerCNN3 as the final model as it achieved the best metrics.

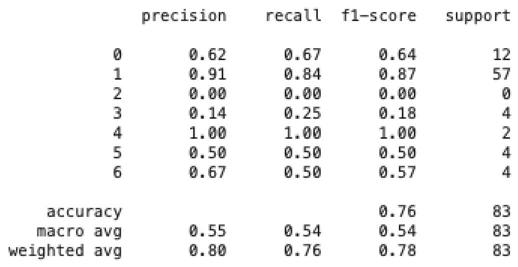
Team 38's final model met expectations, with the following metrics quantifying its success:

As the dataset was duplicated and balanced, model bias will not be of concern when training, thus, it is valid to use accuracy as our choice of performance metric over precision as precision is used for unbalanced datasets. While the F1-score could be used, it is not necessary in this case.

Performance Metrics	Values	Importance
Loss	0.3365	Indicates that the model's predictions are closely matching the actual data, it is close to ideal performance.
Train Accuracy	91.9%	Indicates that the model was able to learn well from the data it was trained on and shows it learns from training patterns.
Validation Accuracy	85.20%	Model performed well on unseen data during training.
Test Accuracy	78.04%	Model performed well on unseen data after training was completed. This metric ensures how ready the model is for real-world deployment.

Table 1: Performance Metrics

The above values for the metrics, primarily the Test Accuracy, indicate that the model has reached a substantial potential to detect skin cancers. However, a test accuracy of 78% is not substantial enough for real-world deployment. Overall, Team 38 has achieved the targeted Test Accuracy of around 80% as it aligns with existing models (further discussed in Section 10).


```
              precision    recall  f1-score   support

           0       0.62      0.67      0.64        12
           1       0.91      0.84      0.87        57
           2       0.00      0.00      0.00         0
           3       0.14      0.25      0.18         4
           4       1.00      1.00      1.00         2
           5       0.50      0.50      0.50         4
           6       0.67      0.50      0.57         4

    accuracy                           0.76        83
   macro avg       0.55      0.54      0.54        83
weighted avg       0.80      0.76      0.78        83
```

Figure 6: Primary Model Classification Report

Figure 6 displays other metrics such as precision, recall, f1-score, and support. These metrics are primarily used for an evaluation of a model's performance in cases of unbalanced data.

8 Qualitative Results

The model's main function is to identify different types of skin lesions, with predictions illustrating its performance. While skin lesions may appear as scars or marks to the untrained eye, the model correctly identifies them 78% of the time.

Figure 7 demonstrates the model's accuracy in predicting 2 out of 3 samples, particularly excelling in class 1 (Basal Cell Carcinoma-BCC) and other classes with fewer original images, thanks to upscaling and augmentation.

Additionally, as shown in Figure 8, after 10 epochs, the model began to overfit the training data, causing validation accuracy to plateau. To address this, increasing the batch size and introducing new data could enhance performance.

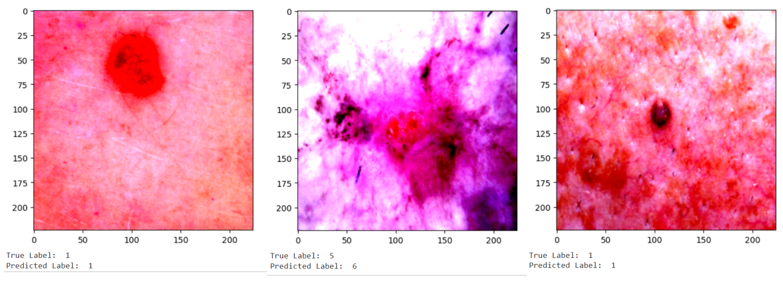


Figure 7: Prediction done by model on Lesions

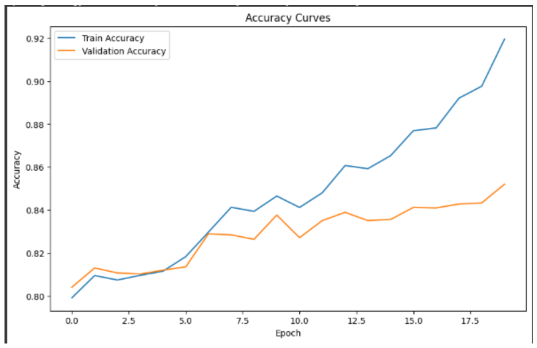
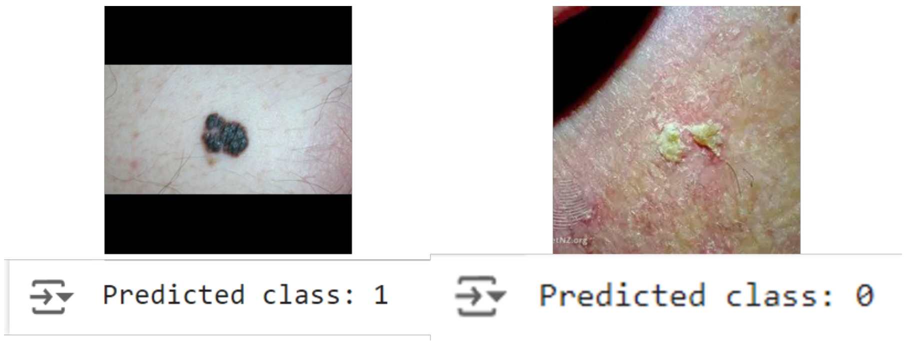


Figure 8: Accuracy Curves

9 Model Evaluation

The model performs on new data found online from medical pages, but it struggles with accuracy when the images have varying lighting or quality. In the figure below, the model correctly predicted the image on the left labeled as class 0 (Actinic Keratosis) but failed to identify the Melanoma image on the right (class 4). Since most online images of skin lesions have already been used to train the model, finding truly new examples is challenging, causing the model's accuracy to drop below its 78% benchmark in real-world settings. These results are expected given the scale and training of the model.



```
Predicted class: 1
```

```
Predicted class: 0
```

Figure 9: Real World Prediction of Model

10 Results Discussion

The model performed as expected given the limited hardware resources, achieving approximately 78% accuracy in predicting skin lesions. The HAM10000 dataset, designed to challenge researchers worldwide, highlighted the relative success of our model, especially considering that only three groups—comprising universities, medical researchers, and CNN experts—achieved over 90% accuracy.

Data augmentation during preprocessing significantly boosted the model's performance. Without transformations, the model struggled to reach 80% validation accuracy, even with high epochs. However, with noisy input images, validation accuracy improved to 83% within 5 epochs, and the loss value dropped from 1.95 to 0.3365 by the final epoch. The architecture began to overfit after 15 epochs, suggesting that a deeper model could achieve better results with the current hyperparameters.

Through training, we learned the importance of balancing the number of layers. SkinCancerCNN2, with 16 layers, overfitted, while SkinCancerCNN3, with 13 layers, outperformed it. This experience underscored the need for a balanced architecture to avoid underfitting and overfitting.

Overall, Team 38's model performed well, achieving a final validation accuracy of 85% and a test accuracy of about 78%, indicating effective performance across various skin lesion types with a relatively low rate of misclassification.

11 Ethical Considerations

Ethical considerations are crucial when developing an AI model for detecting skin issues to ensure the safety and well-being of users.

11.1 Bias in training data

Bias in the training data can pose a significant issue. If the model is trained on a dataset predominantly featuring a specific skin color, such as brown skin, it may not generalize well to other skin colors, such as black or white. This lack of diversity in training data can lead to inaccurate results and pose a significant health risk to users whose skin color is underrepresented, potentially resulting in severe health complications.

11.2 Risk of false diagnosis

There is a risk of false diagnosis results if the model is not properly trained. If the model misinterprets a concerning mark on the skin as benign, it can lead to serious health problems for users who rely on its accuracy. Issues that should be flagged as health concerns may be overlooked and left untreated, thereby increasing the patient's health risk. Inadequate training of the AI model can result in the misinterpretation of specific skin issues, posing a significant threat to human health.

12 Project Difficulty and Quality

The project presents a significant challenge due to the complexity of skin cancer detection, which requires extensive datasets to train a deep learning model effectively and minimize detection errors. A major hurdle is the GPU hardware limitations, which necessitated careful selection of batch size, image size, and CNN layers to optimize GPU utilization within the constraints of free GPU time on Google Colab. The team also explored various techniques to enhance accuracy, such as adaptive learning rates, though these were not particularly effective in our case. Image classification is prone to overfitting if the model isn't sufficiently deep, requiring considerable time to determine the appropriate network architecture, as detailed in Section 5. The model implementation was time-consuming and tedious, with training taking hours due to limited hardware resources. Consequently, the achieved accuracy reflects these constraints, and with more computational power, a deeper, more accurate network could have been trained.

13 Link to Google Colab

Google Colab:
`https://colab.research.google.com/drive/1n6IK9tg4qeSpRf4MuXT1p4Azkb4_O80Y?usp=sharing`

References

[1] M. S. Akter, H. Shahriar, S. Sneha, and A. Cuzzocrea, "Multi-class skin cancer classification architecture based on deep convolutional neural network," in *2022 IEEE International Conference on Big Data (Big Data)*, pp. 5404–5413, 2022.

[2] F. Grignaffini, F. Barbuto, L. Piazzo, M. Troiano, P. Simeoni, F. Mangini, G. Pellacani, C. Cantisani, and F. Frezza, "Machine learning approaches for skin cancer classification from dermoscopic images: A systematic review," *Algorithms*, vol. 15, no. 11, 2022.

[3] S. B. Mukadam and H. Y. Patil, "Skin cancer classification framework using enhanced super resolution generative adversarial network and custom convolutional neural network," *Applied Sciences*, vol. 13, no. 2, 2023.

[4] S. Aladhadh, M. Alsanea, M. Aloraini, T. Khan, S. Habib, and M. Islam, "An effective skin cancer classification mechanism via medical vision transformer," *Sensors*, vol. 22, no. 11, 2022.

[5] P. Bansal, R. Garg, and P. Soni, "Detection of melanoma in dermoscopic images by integrating features extracted using handcrafted and deep learning models," *Computers Industrial Engineering*, vol. 168, p. 108060, 2022.

[6] P. Tschandl, C. Rosendahl, and H. Kittler, "The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions," *Scientific Data*, vol. 5, p. 180161, 2018.

[7] "Easiest way to download kaggle data in google colab," 2024. Accessed: Jul. 4, 2024.

(1) (2) (3) (4) (5)