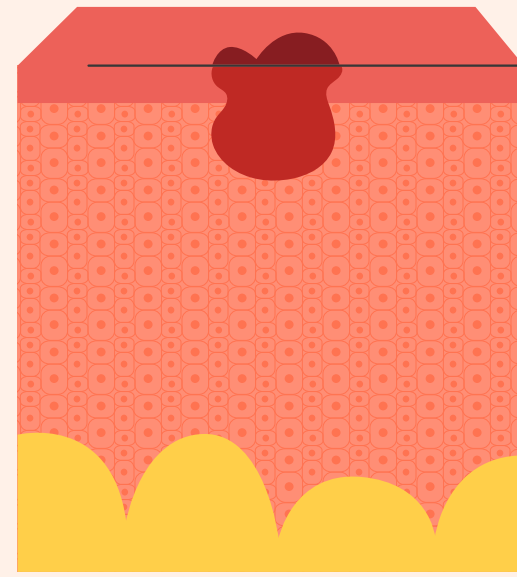
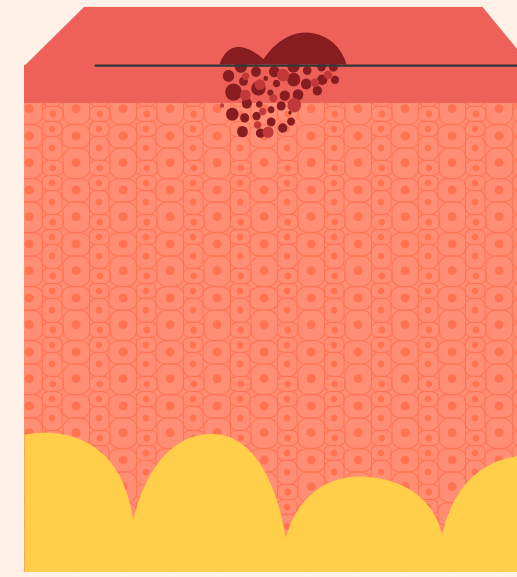


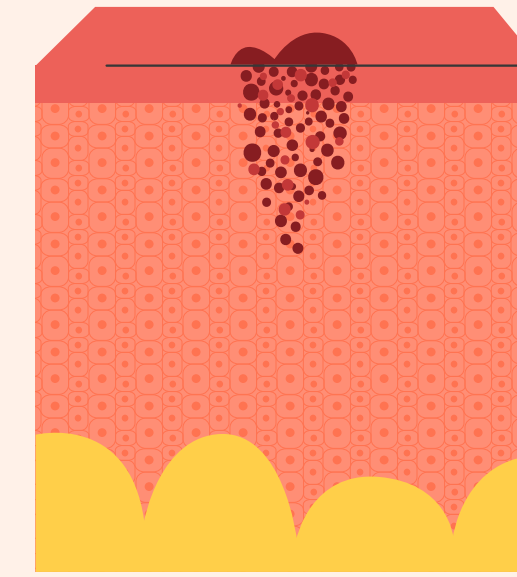
**Stage 0**



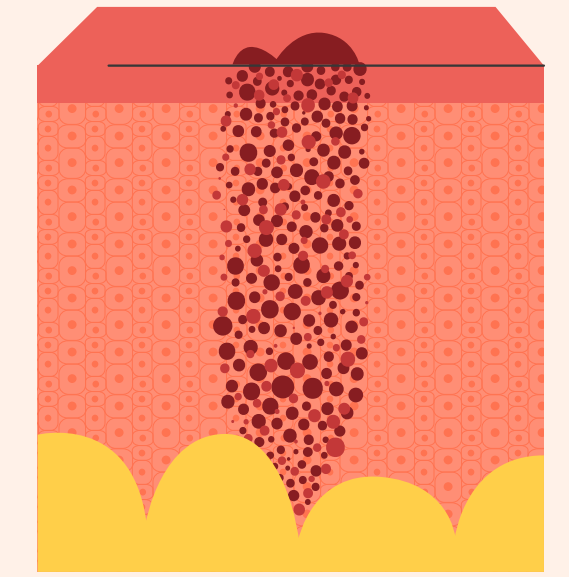
**Stage I**



**Stage II**



**Stage III**



**Stage IV**

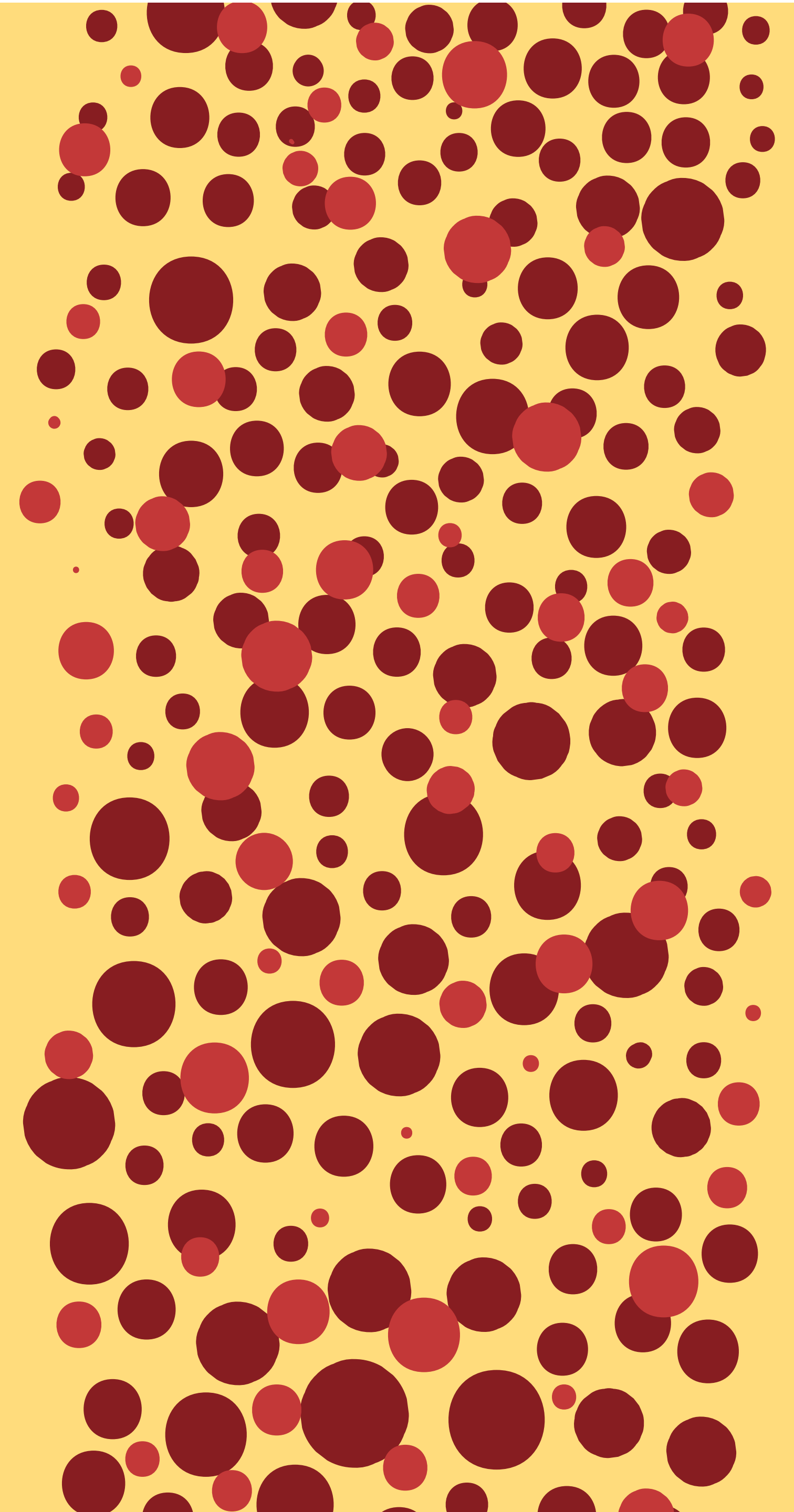
# Skin Lesion Classification Project

Team Name: Team Decision Makers

Team Members: Jesse Kranyak, Siriesha Mandava, Mohamed Altoobli, Jeffery Boczkaja

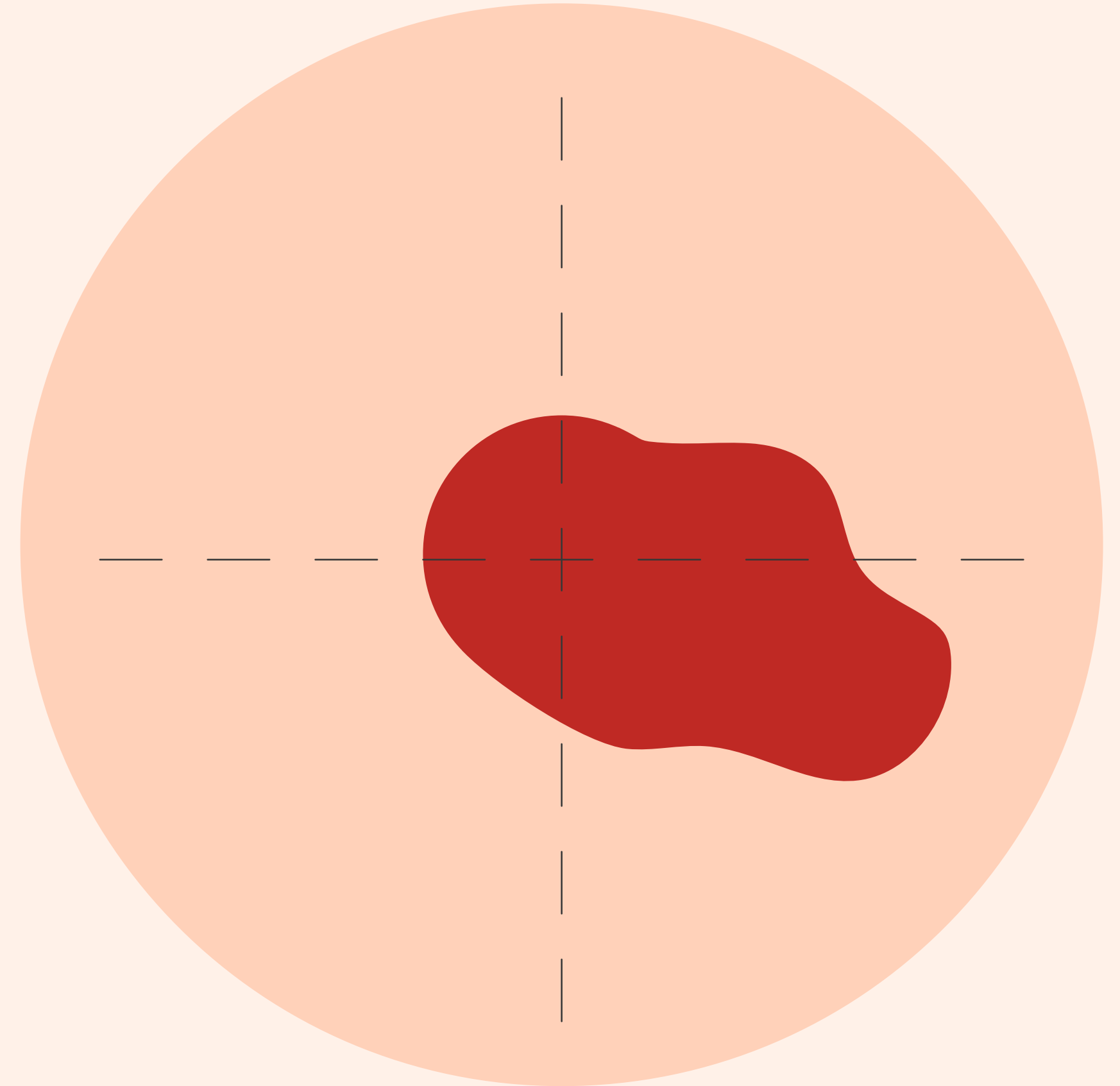
## Introduction

- ▶ **Introduce the problem:** Skin lesions are often indicative of various dermatological conditions, and accurate classification is crucial for timely treatment and management.
- ▶ **Importance:** Early detection of skin diseases can significantly improve patient outcomes, reducing morbidity and mortality rates.
- ▶ **Brief overview of the project goals and objectives:** Develop a deep learning model to classify skin lesions using image data and patient metadata.



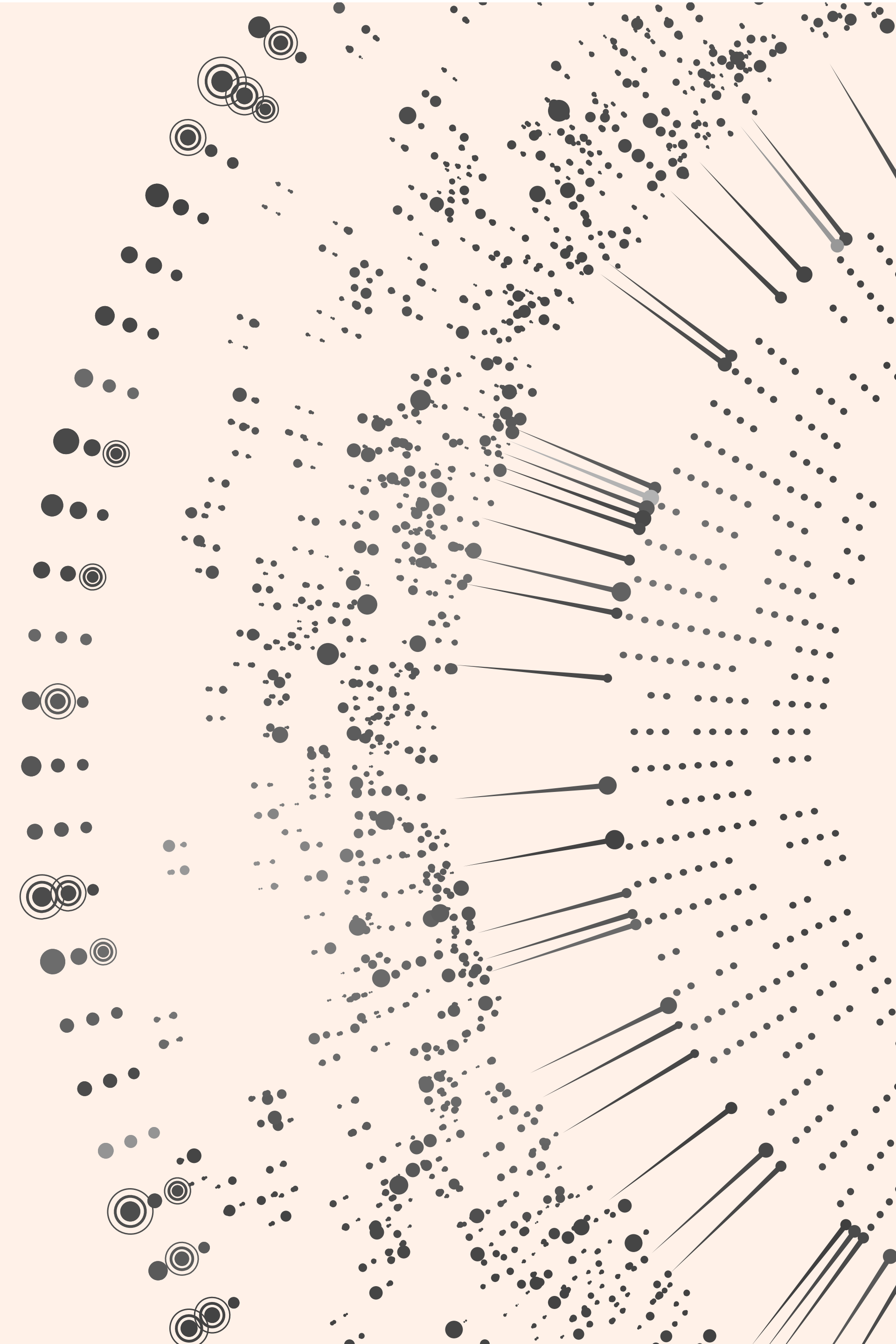
## Project Overview

- ▶ **Description of the ISIC 2019 dataset:**  
A comprehensive dataset containing images of various skin lesions, along with associated metadata such as age, sex, and anatomical site.
- ▶ **Objective:**  
Develop a dual-input neural network model that integrates both image data and patient metadata to improve diagnostic accuracy.
- ▶ **Application:**  
The model aims to serve as a diagnostic aid for dermatologists, facilitating more accurate and efficient diagnosis of skin lesions.



## Data Acquisition

- ▶ **Querying the ISIC API for dataset access:**  
accessing the ISIC 2019 dataset through the API, including authentication and endpoint usage.
- ▶ **Handling metadata for training and testing data:**  
providing context for the images and ensuring proper alignment between image data and patient information.
- ▶ **Preprocessing steps to ensure data readiness for analysis:**  
data preprocessing steps such as resizing images, normalizing pixel values, and handling missing metadata.



## Model Architecture

- ▶ **the dual-input neural network architecture:**  
Break down the architecture into its components, including the image input branch, metadata input branch, and combining branches.
- ▶ **Integration of image data and metadata for comprehensive analysis:**  
Discuss how the model leverages both image data and patient metadata to improve diagnostic accuracy and provide more contextually informed predictions.
- ▶ **Visualization of the model's design:**  
Provide visual representations of the model architecture, including diagrams or flowcharts illustrating the flow of data through the network.





## Training Strategies

- ▶ **Initialization of data generators for training, validation, and testing:**  
the role of data generators in feeding data batches to the model during training, validation, and testing phases.
- ▶ **Techniques such as early stopping and model checkpointing:**  
how early stopping prevents overfitting by halting training when validation loss ceases to improve, and how model checkpointing saves the best model weights based on validation performance.
- ▶ **Optimization methods like hyperparameter tuning:**  
the use of hyperparameter tuning techniques such as grid search or random search to optimize model performance.



## Model Evaluation

### ▶ **Evaluation metrics including accuracy, precision, recall, and F1-score:**

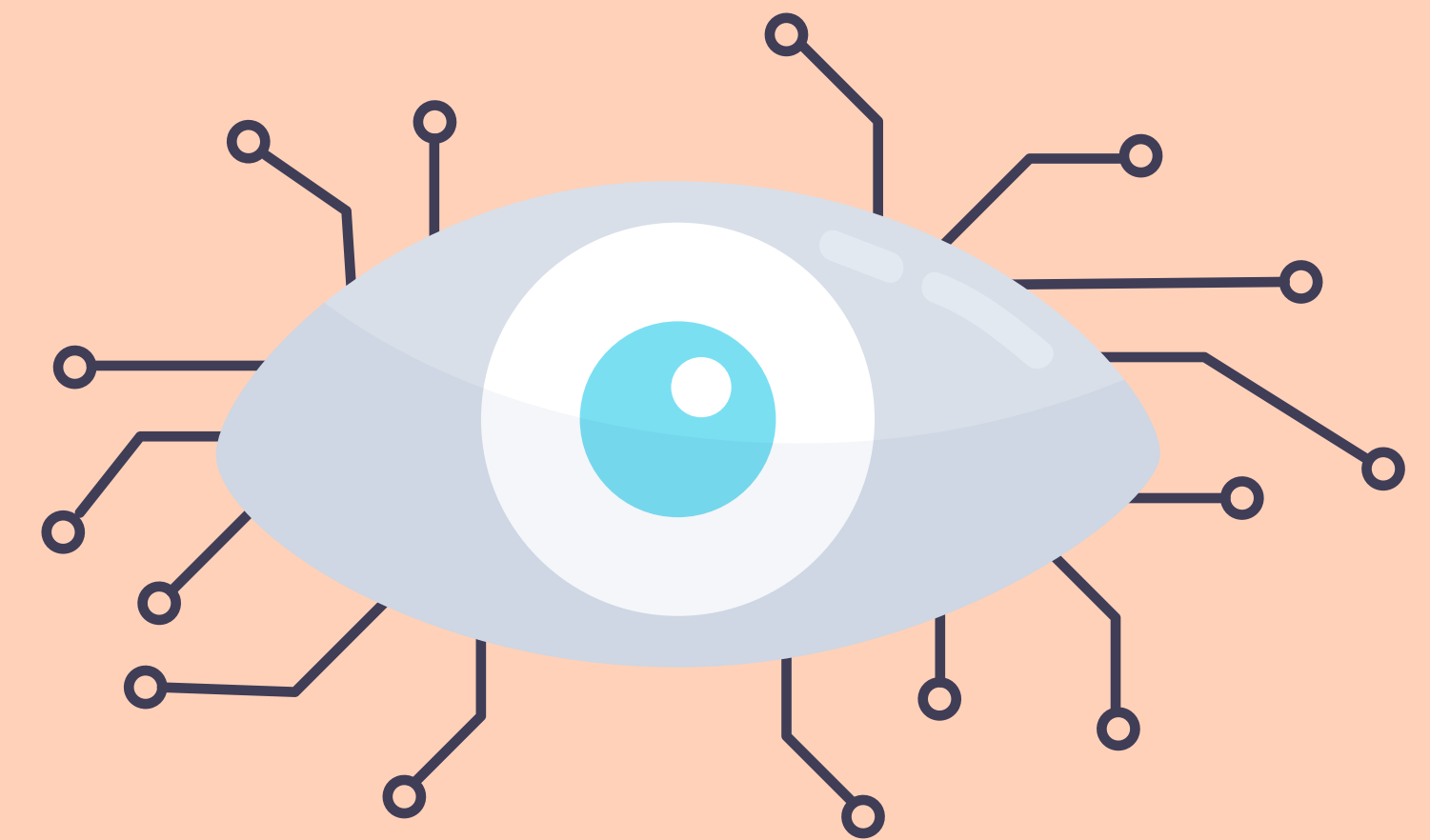
Define each evaluation metric and explain their importance in assessing model performance.

### ▶ **Confusion matrix analysis to identify model performance across different classes:**

examples of confusion matrices and explain how they help identify misclassifications and model weaknesses.

### ▶ **ROC curves and AUC values for per-class evaluation:**

Explain the use of ROC curves and AUC values to assess the model's ability to discriminate between classes and visualize its performance.



## User Interface Development

- ▶ **Development of a web-based user interface using Gradio:**  
the process of developing a user-friendly interface for the model using the Gradio library.
- ▶ **Upload functionality for skin lesion images:**  
users can upload skin lesion images through the interface for prediction.
- ▶ **Integration of metadata input for enhanced predictions:**  
the inclusion of metadata input fields in the interface to provide additional context for predictions.





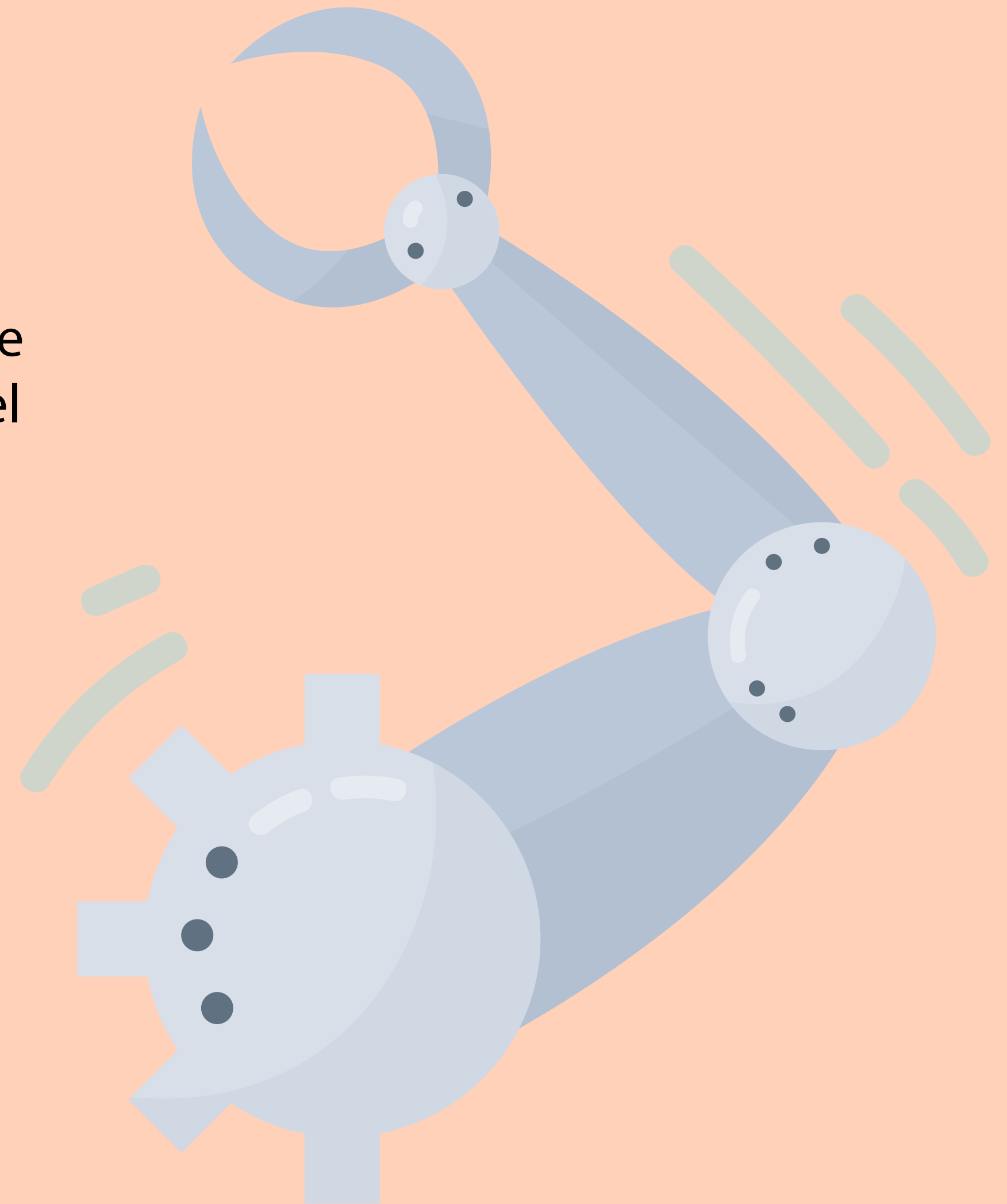
## Knowledge Distillation and Fine-tuning

### ▶ **Techniques for model refinement and improvement:**

knowledge distillation transfers knowledge from a larger, more complex "teacher" model to a smaller, simpler "student" model to improve generalization.

### ▶ **Fine-tuning with regularization for robust feature learning:**

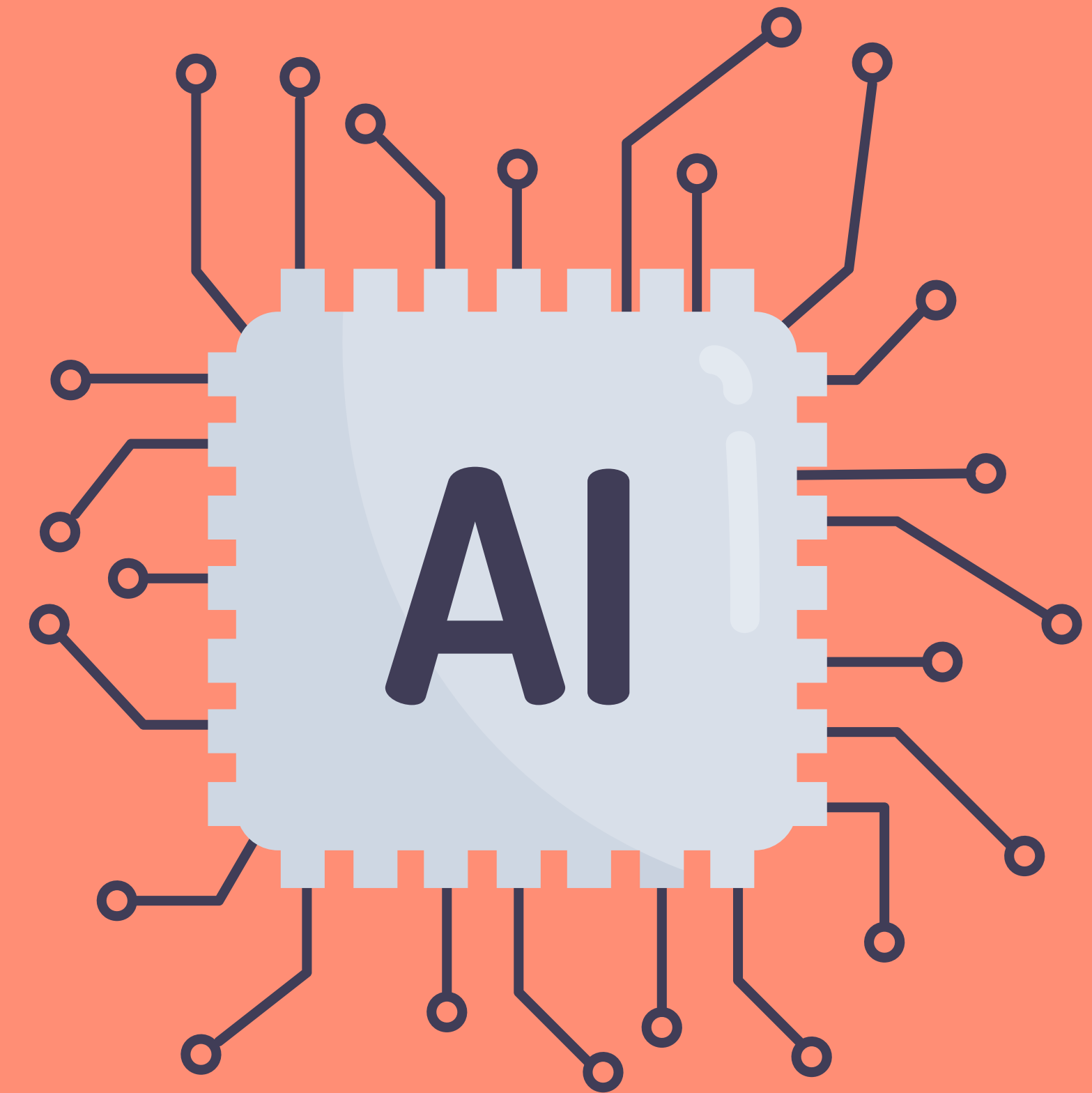
the process of fine-tuning a pre-trained model with regularization techniques such as L1/L2 regularization to prevent overfitting and improve model robustness.



## **Further Model Evaluation**

▶ **Analysis of misclassifications and model weaknesses:**

▶ **Exploration of potential enhancements for model performance:**



## **Future Directions**

### **Potential directions for future development and research:**

possible avenues for expanding the model's capabilities, such as including more diagnostic categories or integrating with electronic medical records systems.

### **Expansion of the model to include more diagnostic categories:**

the model could be extended to classify a broader range of skin lesions, potentially improving its utility in clinical practice.

### **Integration with medical systems for real-world application:**

the feasibility of integrating the model into existing medical systems for use by healthcare providers in real-world settings.

## **Use Case: Diagnostic Aid**

### **Demonstration of the model's application as a diagnostic aid:**

the model could be used by dermatologists to assist in diagnosing skin lesions, potentially improving diagnostic accuracy and patient outcomes.

## **Limitations and Challenges**

- ▶ **Discussion of limitations such as dataset biases and model interpretability:**  
potential biases in the dataset and challenges related to interpreting model predictions in clinical practice.
- ▶ **Challenges encountered during model development and deployment:**  
difficulties encountered during the project, such as data preprocessing issues or technical challenges with model training.





Credits

References

# CancerNet-SCa

- ▶ Introduction: CancerNet-SCa is a deep neural network tailored for skin cancer detection from dermoscopy images.
- ▶ Motivation: Skin cancer is prevalent in the U.S., urging for effective early detection methods.
- ▶ Development: Inspired by deep learning advancements, CancerNet-SCa is developed under the Cancer-Net initiative.



## Key Features

### ▶ **Model Design:**

First machine-designed neural network for skin cancer detection.

### ▶ **Innovations:**

Incorporates attention condensers for improved accuracy.

### ▶ **Performance:**

Outperforms ResNet-50 with superior accuracy and reduced complexity.



## **Open Source & Impact**

- ▶ **Open Source:**  
Available to researchers and clinicians for further development.
- ▶ **Encouragement:**  
Aims to catalyze advancements in skin cancer detection methods.
- ▶ **Future Outlook:**  
Promises a new era of precision diagnostics for dermatological conditions.

```
 dropout (Dropout)          (None, 64)          0      ['batch_normalization[0][0]']
 dense_2 (Dense)             (None, 1)           65      ['dropout[0][0]']

=====
Total params: 11965953 (45.65 MB)
Trainable params: 11965825 (45.65 MB)
Non-trainable params: 128 (512.00 Byte)

Epoch 1/10
642/642 [=====] - ETA: 0s - loss: 0.2005 - accuracy: 0.9240
Epoch 1: val_accuracy improved from -inf to 0.98904, saving model to CancerNet_best_model.keras
642/642 [=====] - 565s 868ms/step - loss: 0.2005 - accuracy: 0.9240 - val_loss: 0.07
Epoch 2/10
642/642 [=====] - ETA: 0s - loss: 0.4416 - accuracy: 0.8318
Epoch 2: val_accuracy did not improve from 0.98904
642/642 [=====] - 542s 844ms/step - loss: 0.4416 - accuracy: 0.8318 - val_loss: 0.44
Epoch 3/10
642/642 [=====] - ETA: 0s - loss: 0.4197 - accuracy: 0.8374
Epoch 3: val_accuracy did not improve from 0.98904
642/642 [=====] - 530s 826ms/step - loss: 0.4197 - accuracy: 0.8374 - val_loss: 0.42
Epoch 4/10
642/642 [=====] - ETA: 0s - loss: 0.4239 - accuracy: 0.8357
Epoch 4: val_accuracy did not improve from 0.98904

[41] # !cp CancerNet_best_model.keras /content/drive/MyDrive/

[39] from google.colab import files

✓ 4s completed at 6:47 PM
```

✓  
40s

```
# Evaluate the model on the test data
test_loss, test_accuracy = model.evaluate(test_metadata_gen, verbose=1)

print("Test Loss:", test_loss)
print("Test Accuracy:", test_accuracy)
```

➞ Found 2534 validated image filenames belonging to 2 classes.  
80/80 [=====] - 37s 459ms/step - loss: 0.4270 - accuracy: 0.8299  
Test Loss: 0.42697906494140625  
Test Accuracy: 0.8299131989479065

[ ]

✓  
6s

```
[42] !pip install -q -U keras-tuner
```

129.1/129.1 kB 4.1 MB/s eta 0:00:00

✓  
2m

```
[43] # Hyperparameter Tuning
import kerastuner as kt
from tensorflow import keras
from kerastuner import Hyperband
from kerastuner import HyperParameters
from tensorflow.keras import layers
from tensorflow.keras.optimizers import Adam
```

✓ 4s completed at 6:47 PM





Found 2051/ validated image filenames belonging to 2 classes.  
Found 2280 validated image filenames belonging to 2 classes.  
Model: "model"

| Layer (type)                              | Output Shape          | Param #  | Connected to                        |
|---|-----------------------|----------|-------------------------------------|
| =====                                     |                       |          |                                     |
| input_1 (InputLayer)                      | [(None, 224, 224, 3)] | 0        | []                                  |
| conv2d (Conv2D)                           | (None, 222, 222, 32)  | 896      | ['input_1[0][0]']                   |
| max_pooling2d (MaxPooling2D)              | (None, 111, 111, 32)  | 0        | ['conv2d[0][0]']                    |
| conv2d_1 (Conv2D)                         | (None, 109, 109, 64)  | 18496    | ['max_pooling2d[0][0]']             |
| max_pooling2d_1 (MaxPooling2D)            | (None, 54, 54, 64)    | 0        | ['conv2d_1[0][0]']                  |
| input_2 (InputLayer)                      | [(None, 5)]           | 0        | []                                  |
| flatten (Flatten)                         | (None, 186624)        | 0        | ['max_pooling2d_1[0][0]']           |
| dense (Dense)                             | (None, 32)            | 192      | ['input_2[0][0]']                   |
| concatenate (Concatenate)                 | (None, 186656)        | 0        | ['flatten[0][0]',<br>'dense[0][0]'] |
| dense_1 (Dense)                           | (None, 64)            | 11946048 | ['concatenate[0][0]']               |
| batch_normalization (Batch Normalization) | (None, 64)            | 256      | ['dense_1[0][0]']                   |

✓ 4s completed at 6:47 PM







