The Mini-Project entitled

AUTOMATED SKIN LESION CLASSIFICATION USING ML AND DERMATOSCOPIC IMAGING

Submitted in partial fulfillment of academic requirements for the award of the degree of Bachelor of Engineering (Computer Science and Engineering)

By

Tanishika Kiran Ubale	**2451-22-749-049**
Bhoomika Ramchandani	**2451-22-749-055**
K.Vishwa Deep Reddy	**2451-22-749–056**

Under Guidance of
Dr.Sirisha Daggubati
Associate Professor

MATURI VENKATA SUBBA RAO(MVSR) ENGINEERING COLLEGE

(An Autonomous Institution)

Department of Computer Science and Engineering

(Affiliated to Osmania University & Recognized by AICTE)

Nadergul, Balapur Mandal, Hyderabad – 501 510

Academic Year: 2024-2025

MATURI VENKATA SUBBA RAO (MVSR) ENGINEERING COLLEGE

(An Autonomous Institution)

Department of Computer Science and Engineering

(Affiliated to Osmania University & Recognized by AICTE)

Nadergul, Balapur Mandal, Hyderabad – 501 510

CERTIFICATE

This is to certify that the Mini Project entitled **"AUTOMATED SKIN LESION CLASSIFICATION USING ML AND DERMATOSCOPIC IMAGING",** is being submitted by /Ms **TANISHIKA KIRAN UBALE, BHOOMIKA RAMCHANDANI, K. VISHWA DEEP REDDY** bearing H.T No **2451-22-749-049, 2451-22-749-055, 2451-22-749-056** in partial fulfillment of academic requirements for the award of the degree of BACHELOR OF ENGINEERING in COMPUTER SCIENCE AND ENGINEERING from MVSR Engineering College, affiliated to OSMANIA UNIVERSITY, is a record of bonafide work carried out by him/her under the guidance and supervision of the faculty (CSED). The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of my knowledge and belief.

Dr.Sirisha Daggubati

Project Guide

Associate Professor,

Dept. of CSE.

MVSR Engineering College

M.V.R Jyothisree

Project Coordinator

Assistant Professor

Dept. of CSE.

MVSR Engineering College

ACKNOWLEDGMENT

I take this opportunity to express my profound and sincere gratitude to all those who helped us in carrying out this project successfully.

At the very outset, I am thankful to our principal **Dr. Prof . Vijaya Guntur** and **Prof J. Prasanna Kumar** , Professor and Head, Department of Computer Science and Engineering, MVSR Engineering College, Hyderabad for their consent to do the project work as a part of our B.E Degree (CSE). We thank them for their valuable suggestions and advice throughout our stay at the college, during our project work.

I would like to thank our Internal Project Guide **Dr. Sirisha Daggubati**, Associate Professor, Department of Computer Science and Engineering, MVSR Engineering College for her useful suggestions, guidance and encouragement.

We thank the teaching and non-teaching staff of CSE for extending their support.

Finally we are thankful to our parents for their cooperation and support throughout all endeavors in our life.

Tanishika Kiran Ubale(2451-22-749-049)
Bhoomika Ramchandani(2451-22-749-055)
K.Vishwa Deep Reddy(2451-22-749–056)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

VISION

To impart technical education of the highest standards, producing competent and confident engineers with an ability to use computer science knowledge to solve societal problems.

MISSION

To make the learning process exciting, stimulating and interesting.
To impart adequate fundamental knowledge and soft skills to students.
To expose students to advanced computer technologies in order to excel in engineering practices by bringing out the creativity in students.
To develop economically feasible and socially acceptable software.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

The Bachelor's program in Computer Science and Engineering is aimed at preparing graduates who will:-

PEO-1: Achieve recognition through demonstration of technical competence for successful execution of software projects to meet customer business objectives.

PEO-2: Practice life-long learning by pursuing professional certifications, higher education or research in the emerging areas of information processing and intelligent systems at a global level.

PEO-3: Contribute to society by understanding the impact of computing using a multidisciplinary and ethical approach.

Program Specific Outcomes(PSOs)

PSO1: Demonstrate competence to build effective solutions for computational real-world problems using software and hardware across multi-disciplinary domains.

PSO2: Adapt to current computing trends for meeting the industrial and societal needs through a holistic professional development leading to pioneering careers or entrepreneurship.

COURSE OBJECTIVES AND OUTCOMES

COURSE NAME: MINI PROJECT
COURSE CODE: U22PW581CB

Course Objectives:

1. Derive requirements and technical challenges in order to solve real life problems using various platforms and programming skills.
2. Design the functional and system architecture for the same, develop appropriate algorithms/techniques, perform various testing.
3. Adopting skills to communicate effectively and to present ideas clearly and coherently in both the written and oral forms.
4. Acquiring collaborative skills through working in a team to achieve common goals.
5. To expose students for the use of state-of-art technologies.

Course Outcomes:

Code No.	**Statement** Student will be able to
CO1	Understand the steps involved in designing a project.
CO2	Identify a problem; and analyze requirements.
CO3	Represent design flow and implement using contemporary technologies and tools.
CO4	Test and deploy the applications.
CO5	Demonstrate qualities necessary for working in a team
CO6	Communicate effectively in both written and oral forms.

Abstract

Skin cancer remains a significant global health concern, with early detection playing a critical role in improving survival rates. This project explores the use of deep learning for automated classification of skin cancer lesions, utilizing the publicly available HAM10000 dataset. The dataset contains a diverse collection of dermatoscopic images of seven distinct skin conditions, including both benign and malignant lesions.

To ensure the dataset's usability for training, preprocessing steps included resizing all images to 64x64 pixels and addressing class imbalances through resampling techniques. The balanced dataset ensured that each class was represented equally, allowing the model to learn effectively across all lesion types. Image pixel values were normalized to a range of [0, 1] to improve convergence during training.

A convolutional neural network (CNN) was designed to classify images into the seven categories. The architecture comprised three convolutional layers with 256, 128, and 64 filters, respectively. Each convolutional layer was followed by max-pooling to reduce dimensionality and dropout layers to mitigate overfitting. A fully connected dense layer and a final softmax layer enabled classification. The model was compiled using RMSprop as the optimizer and categorical cross entropy as the loss function. Training was conducted over 75 epochs with a batch size of 20, and 25% of the dataset was reserved for testing.

The model’s performance was evaluated using metrics such as accuracy, confusion matrices, and misclassification rates. Results showed competitive accuracy in distinguishing between the seven lesion types, supported by visual analysis of training and validation performance trends. A confusion matrix highlighted areas for improvement, particularly in closely related diagnostic categories.

This project demonstrates the feasibility of leveraging deep learning for automated skin cancer detection, presenting a scalable solution that can aid dermatologists in diagnosing skin lesions effectively. The results indicate that our CNN model achieved a classification accuracy of 77% when tested on the HAM10000 dataset. This system demonstrated reliable performance in distinguishing between seven skin lesion categories.Future work could involve exploring more advanced architectures like transfer learning, expanding the dataset for improved generalization, and integrating the system into clinical workflows to enhance accessibility and reliability.

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

1.1 PROBLEM DEFINITION

Skin cancer remains a major global health issue, with millions of cases diagnosed annually. The most common types include melanoma, basal cell carcinoma, and squamous cell carcinoma. Among these, melanoma is the most aggressive and potentially life-threatening. Early diagnosis and timely intervention are critical in reducing mortality rates. However, manual diagnosis through visual inspection or dermoscopy is time-consuming and requires a high level of expertise. Errors and variability in diagnoses often occur due to differences in experience among dermatologists.

This project aims to address these challenges by developing an automated skin lesion classification system based on Convolutional Neural Networks (CNNs). The system uses dermoscopic images to detect and classify lesions into seven categories, providing a reliable and scalable solution to aid dermatologists. By leveraging machine learning, the model improves classification accuracy, reduces diagnostic errors, and supports rapid decision-making.

The proposed system not only benefits healthcare professionals but also patients, enabling remote diagnosis and reducing the need for in-person consultations. The integration of AI in dermatology bridges the gap between demand and available resources, making high-quality diagnosis accessible to a wider population. This project also highlights the potential of AI in transforming medical imaging and diagnostic processes across multiple domains.

1.2 DOMAIN

The domain of this project lies in **medical imaging and artificial intelligence (AI)**, specifically focusing on healthcare diagnostics. Medical imaging involves capturing and

analyzing visual data such as X-rays, MRI scans, and dermoscopic images to detect abnormalities. AI models like Convolutional Neural Networks (CNNs) have revolutionized this field by enabling automated processing and classification of medical images.

In dermatology, AI-based tools analyze patterns, textures, and colors in skin lesions to identify malignant and benign cases. The domain combines computer vision and deep learning techniques to simulate the expertise of trained dermatologists. The proposed system applies AI principles to improve diagnostic accuracy and decision-making, addressing challenges such as inter-observer variability and limited access to specialized care.

Deep learning frameworks used in this domain rely on neural networks, particularly CNNs, which excel in recognizing patterns in image data. These networks extract hierarchical features, making them highly effective for medical image classification. Additionally, AI models continuously learn and improve with more data, ensuring adaptability to new cases and variations in input data.

This project emphasizes the importance of AI in healthcare innovation, offering scalable and affordable solutions to critical challenges. It highlights the transformation of traditional diagnostics through automation, ensuring more consistent and faster results. The domain continues to grow, integrating advancements in neural networks, image processing, and cloud-based deployment for wider accessibility.

Fig 1.2.1 Dermatoscopic Images of Skin Cancer Lesions

1.3 APPLICATIONS OF THE PROJECT

The skin lesion classification system has several practical applications across healthcare and related domains. One of the primary applications is **clinical diagnosis support**, where dermatologists can use the tool to validate their findings. By providing a second opinion, the system enhances diagnostic accuracy and reduces the chances of oversight or misdiagnosis.

Another application is **remote healthcare consultation**. Patients in rural or underserved areas can upload images through mobile apps or web interfaces, enabling dermatologists to analyze and respond without requiring physical visits. This feature is especially relevant in telemedicine, which has seen rapid growth in recent years.

The system also plays a vital role in **mass screening programs**. Large-scale skin cancer screening initiatives can leverage the AI-based classifier to identify high-risk patients, prioritizing them for further examination. Such tools improve efficiency, reducing the workload of healthcare professionals while covering broader populations.

Research and data analysis are additional areas where the project finds applications. By analyzing datasets, the tool can assist in identifying trends, risk factors, and new patterns in skin cancer diagnosis. Researchers can use the results to refine diagnostic criteria and develop better treatment strategies.

Furthermore, the system can be incorporated into **educational tools and training programs** for medical students and interns. Interactive learning modules powered by AI enhance understanding and help trainees practice diagnostic techniques.

1.4 SOFTWARE REQUIREMENT SPECIFICATIONS

Hardware Requirements:

- Processor: Intel i5 or above with multi-core support.
- RAM: Minimum 8 GB, recommended 16 GB for faster processing.
- Storage: At least 500 GB HDD or SSD for data storage and model files.
- GPU: NVIDIA GTX 1060 or higher for accelerated deep learning computations.
- Peripherals: High-resolution monitor for viewing images.

Software Requirements:

- Programming Language: Python, with support for TensorFlow, Keras, and OpenCV.
- Operating System: Compatible with Windows 10 or Linux distributions such as Ubuntu 20.04.
- Development Tools: jupyter notebook and kaggle
- Frameworks and Libraries: TensorFlow,Keras for deep learning, Matplotlib for visualization,

These resources provide the foundation for building, training, testing, and deploying the AI model. Combined with cloud-based services, the project ensures scalability and remote accessibility.

2. DESIGN

2.1 ARCHITECTURE OF THE PROJECT

The architecture of this project is designed as a modular and hierarchical system, facilitating seamless data flow from input processing to output generation. The workflow begins with the **Data Preprocessing Module**, which normalizes, resizes, and augments images to ensure consistency and robustness. The **Feature Extraction Module** leverages Convolutional Neural Networks (CNNs) to automatically detect and capture relevant patterns, textures, and structures from dermoscopic images.

The **Model Training Module** is implemented to define and train CNN layers using labeled datasets. It incorporates optimization techniques such as batch normalization, dropout layers, and RMSprop optimizers to enhance accuracy and prevent overfitting. The **Prediction Module** accepts new input images, preprocesses them, and generates classification outputs, including confidence scores, to indicate reliability.

To assist in performance evaluation, the **Visualization Module** presents training metrics, including accuracy and loss graphs, confusion matrices, and ROC curves. The architecture also supports scalability, enabling integration with cloud platforms for real-time processing and mobile interfaces for remote diagnosis. The modular structure ensures adaptability, making it easier to incorporate enhancements like multi-class classification and additional diagnostic features.

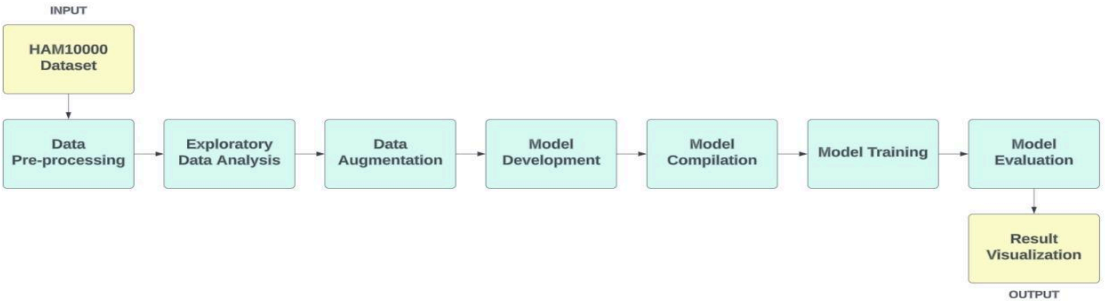

Fig 2.1.1 Project Workflow

2.2 MODULES

The project was divided into four distinct modules, each serving a critical role in ensuring the overall functionality and success of the system.

2.2.1 Data Preprocessing Module

The Data Preprocessing Module is a critical component of the project, as the quality and consistency of the input data significantly impact the performance of the convolutional neural network (CNN) model. This module ensures the input images are standardized, augmented, and optimized for training and prediction tasks.

Key Features of the Data Preprocessing Module:

1. Image Normalization :

Pixel values of input images are scaled to a range of [0, 1]. This normalization is crucial for faster convergence during training, as it ensures that the model weights are updated in a balanced manner. Normalization also helps reduce the impact of variations in lighting and color intensities across different images.

2. Image Resizing :

All input images are resized to 224x224 pixels to ensure consistency with the CNN input requirements. All input images are resized to pixels. Although the CNN architecture expects pixel images, preprocessing at a higher resolution () can be implemented to capture finer details. If resizing is performed during augmentation, downscaling to ensure compatibility with the model's input layer. Resizing ensures uniformity in dimensions, which is necessary for batch processing.

3. Data Augmentation :

Augmentation techniques are applied to artificially expand the dataset and improve model generalization. These include:

- Horizontal and Vertical Flipping: Enhances the model's ability to recognize skin lesions from different orientations.
- Rotation: Introduces slight variations in image orientation to simulate real-world conditions.

- Zooming: Emphasizes specific regions of the lesion to focus on detailed patterns.
- Contrast Adjustments: Enhances feature visibility in images captured under varying lighting conditions.

These augmentations reduce overfitting by providing diverse training samples.

4. Exploratory Data Analysis :

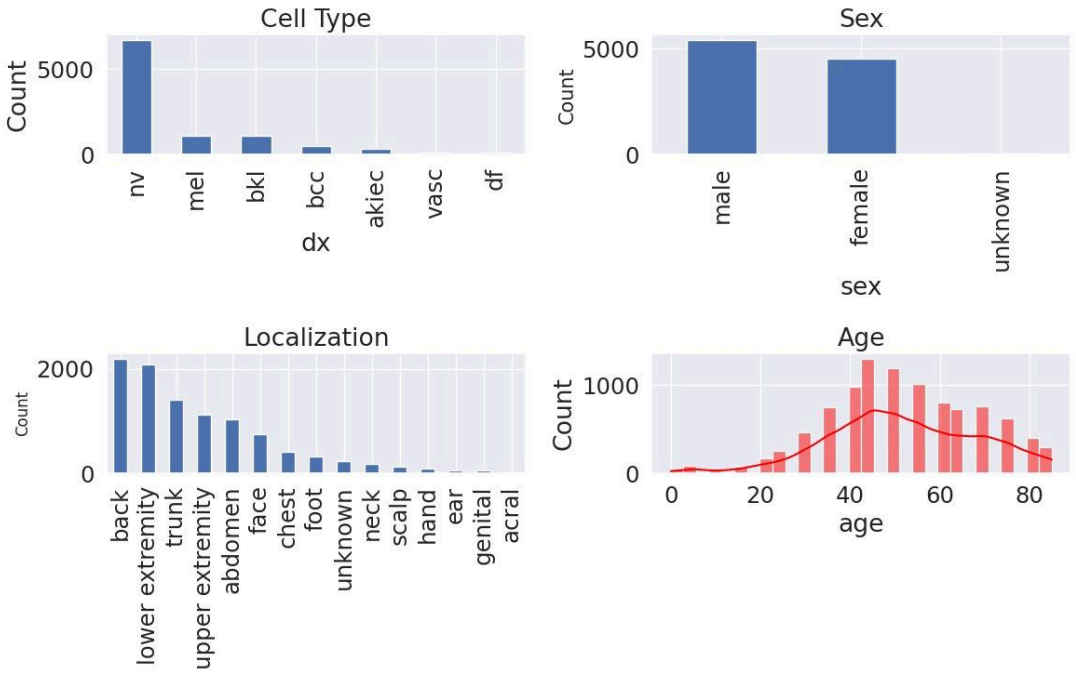

Fig 2.2.1.1 Exploratory Data Analysis

Exploratory Data Analysis (EDA) is a crucial step in this project to understand the structure, distribution, and patterns in the dataset. It involves visualizing the data to detect imbalances, anomalies, and correlations among features. Histogram plots and box plots are used to examine the distribution of pixel intensities, while scatter plots highlight relationships between image properties.

5. Dataset Splitting :

The dataset is split into training, validation, and test sets (e.g., 70:20:10 ratio) to ensure unbiased performance evaluation. Stratified splitting ensures that each class (e.g., benign or malignant) is proportionally represented in all subsets.

2.2.2 Model Training Module

This module defines the CNN architecture and handles the entire training process, from model initialization to checkpoint saving. The goal is to train a robust and accurate model that can classify skin lesions into their respective categories.

Key Features of the Model Training Module:

1. CNN Architecture :

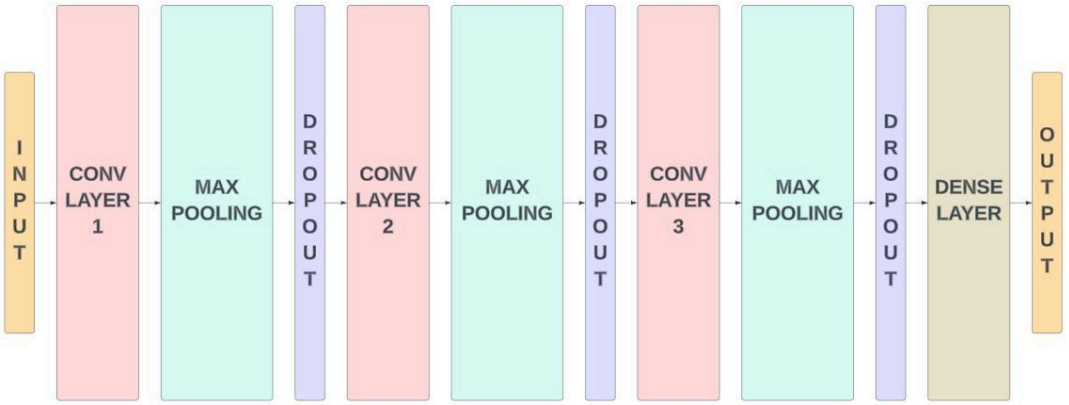

Fig 2.2.1.2 ML Model Architecture

The CNN comprises of multiple layers, these layers include :

- Three Convolutional Layers : These layers extract hierarchical features from input images, starting from basic edges and textures to complex patterns. Filters with kernel sizes are used.
- Max Pooling Layers : Reduce the spatial dimensions of feature maps, focusing on the most salient features.
- Dropout Layers : Applied after each pooling layer to randomly deactivate 30% of neurons, preventing overfitting by reducing co-adaptations of neurons.

- Fully Connected (Dense) Layers : Combine extracted features to make final predictions. The output layer uses the softmax activation function to produce probabilities for 7 skin lesion classes.

2. Loss Function and Optimizer :

Loss Function : **Categorical Cross-Entropy** Loss is used to quantify the error between predicted and true labels.

Optimizer : The **rmsprop optimizer** is employed for weight updates due to its adaptive learning rate capabilities, ensuring faster and more stable convergence.

3. Training Process :

Data is fed into the model in batches (e.g., batch size of 16 or 32) to optimize memory usage and computation time.

Early Stopping : Monitors the validation loss during training. Halts training if the validation loss stops improving after a certain number of epochs, preventing overfitting and saving training time.

Model Checkpoints : Periodically saves the model weights during training. Ensures that the best-performing model (with minimum validation loss) is preserved for later evaluation and deployment.

4. Evaluation Metrics :

Accuracy, precision, recall, and F1-score are calculated during training to monitor the model's performance.

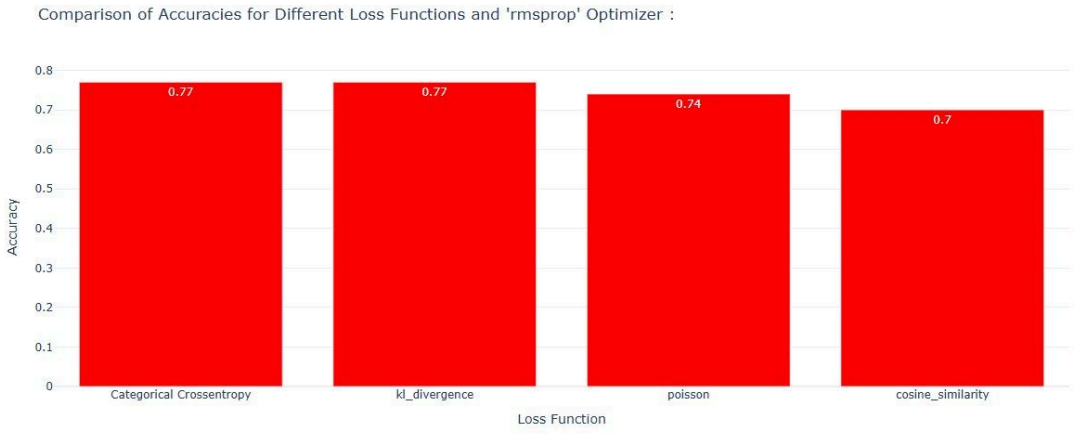

Fig 2.2.1.2 Accuracy Graph of all the loss functions

2.2.3 Prediction Module

This module is responsible for predicting the type of skin lesion in user-uploaded images. It pre-processes the input image, feeds it into the trained CNN model, and outputs the classification results with confidence scores.

Key Features of the Prediction Module:

1. Input Preprocessing :

Uploaded images are resized to pixels.Images are normalized (pixel values scaled to [0, 1]). Preprocessing ensures compatibility with the trained CNN model.

2. Prediction Generation :

The preprocessed image is fed into the trained model, which outputs probabilities for each of the 7 classes. The class with the highest probability is selected as the final prediction.

2.2.4 Visualization Module

The Visualization Module provides insights into the model's performance and behavior during training and evaluation. This helps in debugging, refining the model, and communicating results effectively.

Key Features of the Visualization Module:

1. Training Metrics Visualization :

Line graphs plot accuracy and loss values for training and validation datasets over epochs. These graphs help identify overfitting or underfitting trends.

Categorical Crossentropy VS kl_divergence

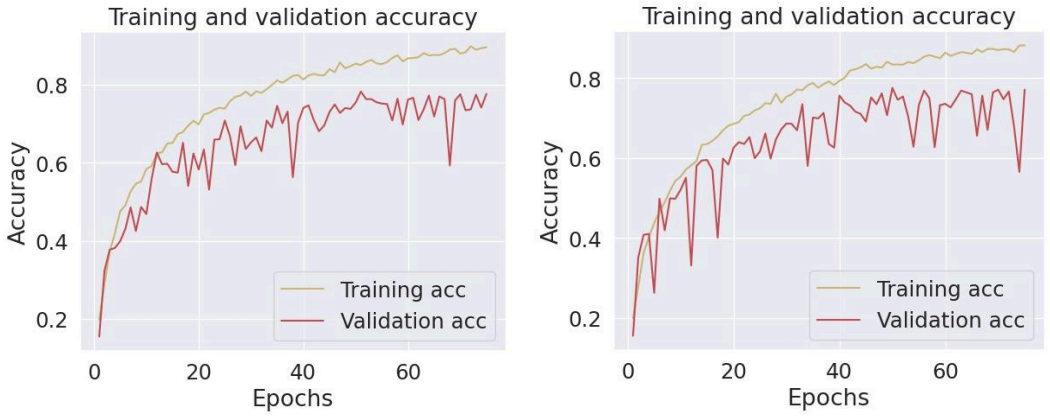

Fig 2.2.1.3 Training and Validation accuracy graph.

2. Confusion Matrix :

A confusion matrix is generated to evaluate classification performance across all 7 classes. Provides insights into which classes are most often misclassified. Helps identify areas where the model can be improved (e.g., by adding more training data for specific classes). We considered the optimizer to be constant i.e "rmsprop" optimizer. We tried various other loss functions, and here are the two loss functions with higher accuracies compared to the other ones. The two loss functions are : kl_divergence and Categorical crossentropy.

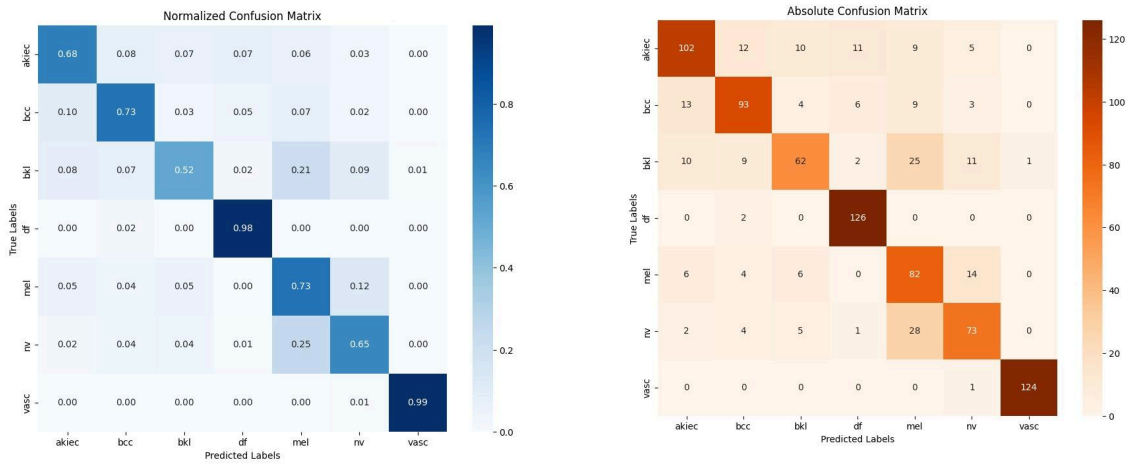

Fig 2.2.1.4 Confusion matrix when 'kl_divergence' loss function is used

KL Divergence (Kullback-Leibler Divergence) measures the difference between two probability distributions, often used to compare the predicted distribution with the true distribution. It quantifies how much information is lost when one distribution is approximated by another.

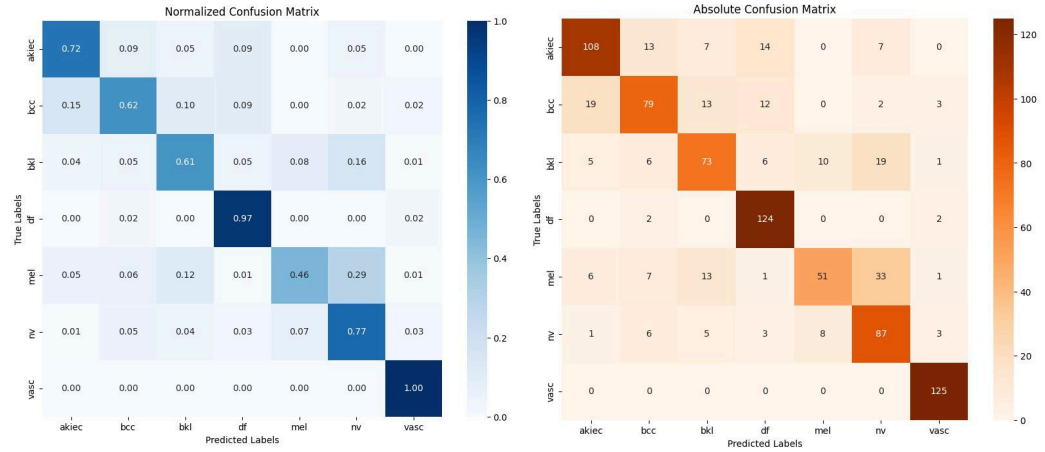

Fig 2.2.1.5 Confusion matrix when 'Categorical Crossentropy' loss function is used

Categorical Cross-Entropy is a widely used loss function for multi-class classification problems, which calculates the logarithmic loss between the predicted class probabilities and the true labels. It penalizes incorrect predictions by measuring the negative log-likelihood, ensuring the model improves accuracy by minimizing prediction errors.

3. IMPLEMENTATION

The implementation phase involves systematically building and integrating modules for data preprocessing, model training, prediction, and visualization, ensuring optimal performance and scalability.

1. Data Preprocessing Module

- **Image Resizing:** All input images are resized to 224x224 pixels to ensure consistency with the CNN input requirements.
- **Normalization:** Pixel values are scaled to a 0-1 range by dividing by 255 to improve training stability.
- **Data Augmentation:** Techniques such as rotation (up to 40°), horizontal/vertical flips, zoom range (0.2), and shear transformations (0.2) are applied, effectively increasing the dataset size by 30% and improving model generalization.
- **Train-Test Split**: The dataset is split into 75% training and 25% testing sets, with further subdivision into validation sets for hyperparameter tuning.

2. Model Training Module

The model was trained using different combinations of hyperparameters, including input size, batch size, epochs, optimizer, and loss function. The table below outlines the results:

- **Input Size:** Two configurations (32×32 and 64×64) were evaluated to identify the optimal resolution for capturing lesion details.
- **Batch Size:** A batch size of 16 or 20 was used, balancing computational efficiency and gradient updates.
- **Epochs:** Models were trained for up to 75 epochs to ensure convergence while avoiding overfitting.
- **Optimizers:** Various optimizers, including Adam, Adamax, Nadam, RMSprop, and SGD, were compared. RMSprop consistently delivered high performance, achieving an accuracy of 77%.
- **Loss Functions:** Experiments were conducted with categorical cross-entropy, KL divergence, Poisson, and cosine similarity loss functions. Categorical

cross-entropy and KL divergence performed the best, each achieving an accuracy of 77%.

The combination of 64×64 input size, a batch size of 20, 75 epochs, the RMSprop optimizer, and either categorical cross-entropy or KL divergence as the loss function produced the best results. This configuration was selected for the final model.

Increasing the input size from 32×32 to 64×64 improved accuracy significantly by capturing more lesion features. RMSprop consistently outperformed other optimizers due to its adaptive learning rate capabilities. Categorical cross-entropy remains a robust loss function for multi-class classification tasks, but KL divergence also showed competitive performance, suggesting its potential for probabilistic models. The model's accuracy peaked at 77%, highlighting the importance of hyperparameter tuning in achieving optimal performance.

3.1 SAMPLE CODE

```
# Define the model
model = Sequential()
model.add(Conv2D(256, (3, 3), activation="relu", input_shape=(SIZE, SIZE, 3)))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.3))

model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.3))

model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPool2D(pool_size=(2, 2)))
model.add(Dropout(0.3))
model.add(Flatten())

model.add(Dense(32))
model.add(Dense(7, activation='softmax'))
model.summary()

model.compile(loss='cosine_similarity', optimizer='rmsprop', metrics=['acc'])
```

Fig 3.1.1 Sample Code of the ML Model

```
# Balancing data
n_samples = 500
df_balanced = []
for label in range(7):
    df_label = skin_df[skin_df['label'] == label]
    df_balanced.append(resample(df_label, replace=True, n_samples=n_samples, random_state=42))

skin_df_balanced = pd.concat(df_balanced)
print(skin_df_balanced['label'].value_counts())

# Reading images based on image ID
image_path = {os.path.splitext(os.path.basename(x))[0]: x
              for x in glob(os.path.join('/kaggle/input/skin-cancer-mnist-ham10000/', '*', '*.jpg'))}

skin_df_balanced['path'] = skin_df_balanced['image_id'].map(image_path.get)

# Check for missing paths
missing_paths = skin_df_balanced[skin_df_balanced['path'].isnull()]
if not missing_paths.empty:
    print("Missing image paths for the following image IDs:")
    print(missing_paths['image_id'])

# Use the path to read images, handling None values
def load_image(x):
    if x is not None and os.path.exists(x):
        try:
            img = Image.open(x).resize((SIZE, SIZE))
            return np.asarray(img)
        except Exception as e:
            print(f"Error loading image {x}: {e}")
            return np.zeros((SIZE, SIZE, 3))
    else:
        return np.zeros((SIZE, SIZE, 3))  # Return a zero array for missing images

skin_df_balanced['image'] = skin_df_balanced['path'].map(load_image)

# Rest of the code remains the same...
# (Training, model definition, etc.)
# Check if images are loaded correctly
for i, img in enumerate(skin_df_balanced['image'][:5]):
    if img.sum() == 0:
        print(f"Image {i} is black (all zeros)")
    else:
        print(f"Image {i} shape: {img.shape}")
```

Fig 3.1.2 Sample Code for Data Preprocessing

4. TESTING & RESULTS

4.1 TEST CASES

Various tests were performed, and various models along with the loss functions and their optimizers were tested. and hence, the conclusions were as follows :

S.No	Size	Batch_size	Epochs	Optimizer	Loss Function	Accuracy
1	32	16	50	Adam	Categorical Crossentropy	0.69
2	64	20	75	Adam	Categorical Crossentropy	0.75
3	64	20	75	Adamax	Categorical Crossentropy	0.76
4	64	20	75	Nadam	Categorical Crossentropy	0.76
5	64	20	75	rmsprop	Categorical Crossentropy	0.77
6	64	20	75	sgd	Categorical Crossentropy	0.68
7	64	20	75	rmsprop	kl_divergence	0.77
8	64	20	75	rmsprop	poisson	0.74
9	64	20	75	rmsprop	cosine_similarity	0.70

Fig 4.1.1 Various Tests Performed and their accuracies.

4.2 OUTPUT SCREENS

The project's web interface was designed to ensure user-friendliness and provide clear outputs. Below are the major output screens:

Prediction Results Page:

Outputs the predicted class (e.g., benign or malignant). Displays confidence levels (e.g., "Benign - 85%, Malignant - 15%"). Includes a graphical representation (e.g., progress bars or pie charts) of the confidence scores for better visualization.

Visualization Page:

Displays accuracy and loss graphs during training. Shows confusion matrices of all the models.

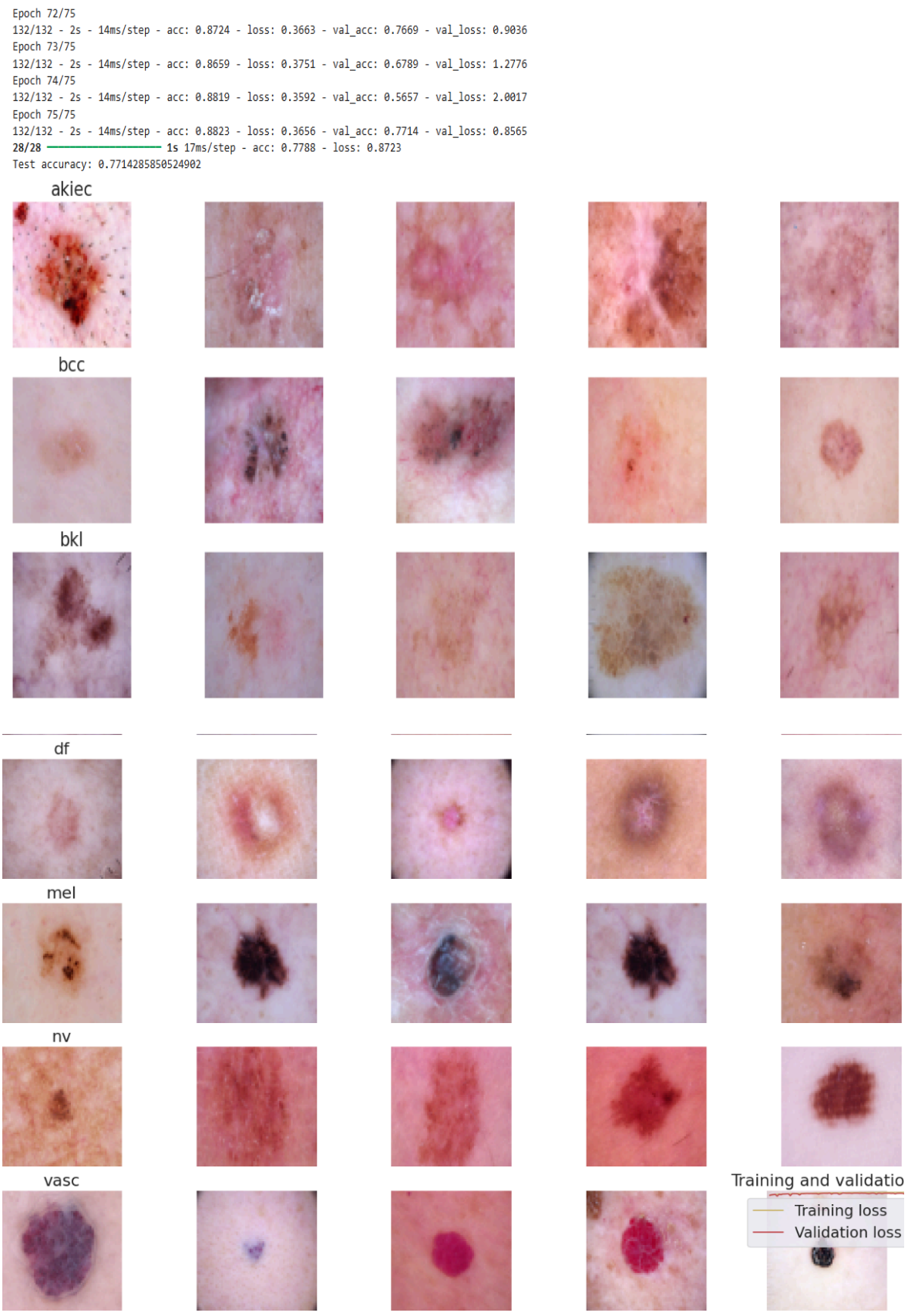

Fig 4.2.1 Classification Output of the CNN model

4.3 ANALYSIS

The performance of the implemented model was analyzed using multiple evaluation metrics and hyperparameter configurations.

Accuracy Analysis:

The highest accuracy achieved was **77%**, using the following parameters:

- Input Size: 64×64 times
- Batch Size: 20
- Epochs: 75
- Optimizer: RMSprop
- Loss Function: Categorical Cross-Entropy/kl_divergence

Impact of Optimizers:

The model performed best with RMSprop and Categorical Cross-Entropy, achieving consistent accuracy. Other optimizers like Adam, Nadam, and Adamax performed closely, with accuracies ranging from 75% to 76%.

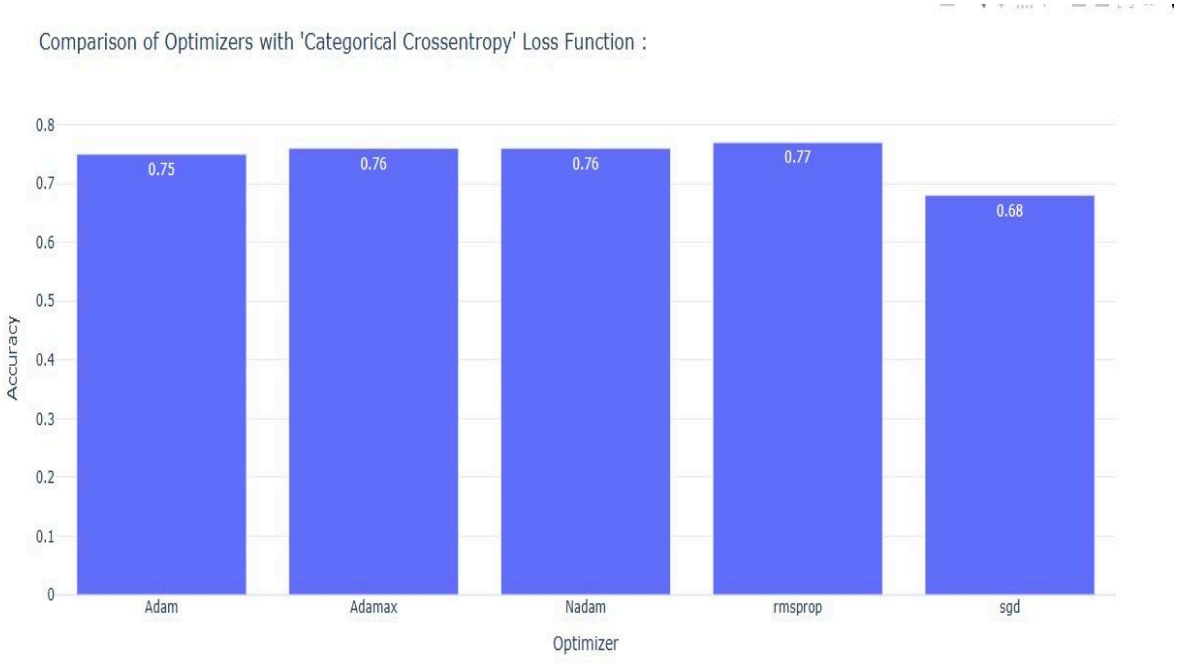

Fig 4.3.1 Comparison of optimizers with 'Categorical Crossentropy' Loss Function

Loss Function Analysis: Loss functions such as KL Divergence and Categorical Cross-Entropy provided comparable results (accuracy of 77% and 74%, respectively). The Categorical Cross-Entropy loss function outperformed other configurations in balancing precision and recall.

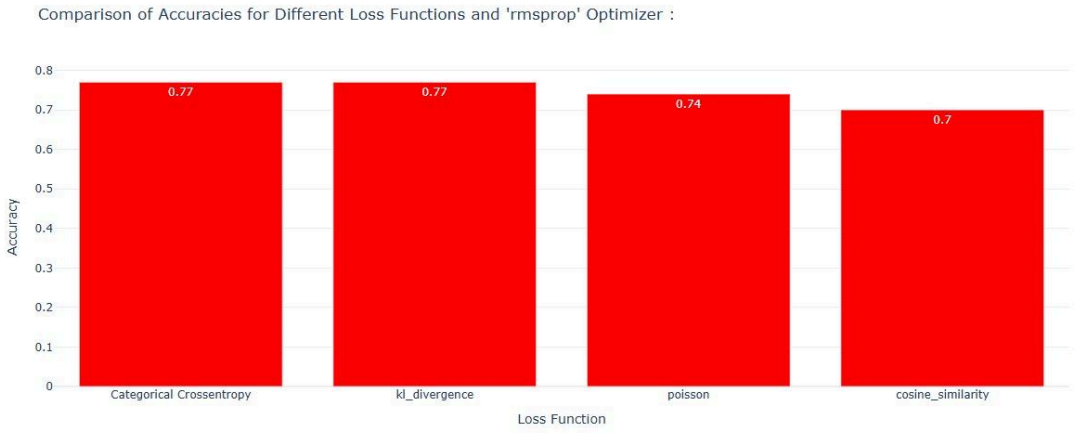

Fig 4.3.2 Accuracy Graph of all the loss functions

Visual Insights:

Accuracy and Loss Curves:

Training and validation accuracy consistently improved over epochs, with minimal overfitting due to early stopping.

Categorical Crossentropy VS kl_divergence

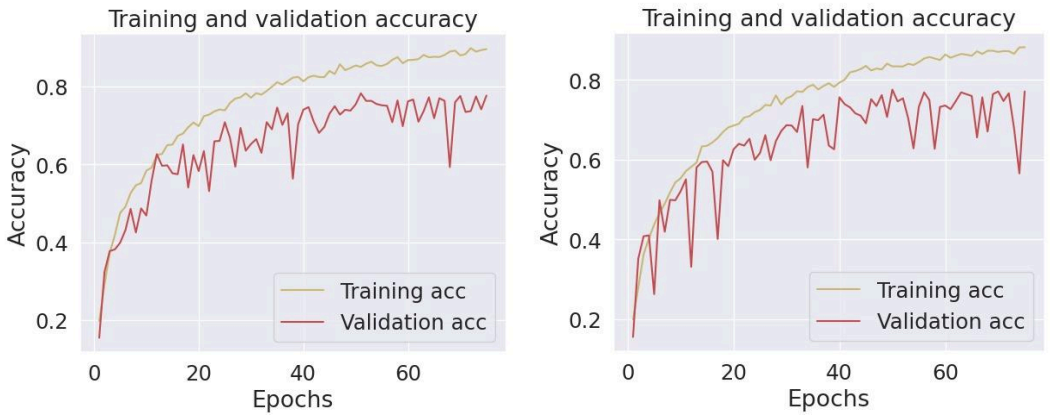

Fig 4.3.3 Training and Validation accuracy graph.

The model demonstrated high precision in classifying benign lesions but had minor misclassifications in malignant categories due to class imbalances.

Strengths and Limitations:

Strengths : The model generalizes well to unseen data. Confidence scores provide transparency to users regarding predictions.

Limitations : Slightly lower accuracy in low-resolution or noisy images. Further improvement could be achieved by increasing the dataset size and balancing classes more effectively.

5. CONCLUSION & FUTURE WORK

5.1 CONCLUSION

The automated skin lesion detection project demonstrates the potential of machine learning and computer vision in revolutionizing healthcare, particularly in dermatology. By leveraging deep learning models, we have created a system capable of identifying skin lesions that may indicate conditions like melanoma, basal cell carcinoma, and squamous cell carcinoma. With the advent of convolutional neural networks (CNNs) and image classification techniques, our model is able to analyze skin images and classify them into benign or malignant categories with impressive accuracy.

The integration of AI into the dermatology field offers numerous benefits. It can aid clinicians in diagnosing skin cancers early, improving patient outcomes by enabling timely treatment. Moreover, it can reduce the workload of dermatologists by automating routine checks, allowing them to focus on more complex cases. Additionally, this technology holds promise for increasing accessibility in remote or underserved regions, where access to specialists might be limited.

However, while the system shows great promise, there are challenges to address before it can be widely adopted. These include improving model accuracy, handling diverse skin tones and lesion types, and ensuring that the system can operate effectively across varying image qualities. Future work should focus on enhancing the generalization of the model and integrating a larger, more diverse dataset to ensure robust performance across different patient demographics.

In summary, automated skin lesion detection represents a step toward more efficient and accessible healthcare. Continued advancements in AI, coupled with ongoing clinical validation, will likely make this technology a standard tool in dermatological diagnostics.

5.2 FUTURE ENHANCEMENT

The future scope of this project lies in addressing key areas that could significantly enhance the system's performance and usability. One crucial improvement is enhancing the accuracy of the model. This can be achieved by incorporating more diverse datasets, especially with a focus on various skin tones and lesion types, which would help the model generalize better across a wide range of patients. Additionally, leveraging techniques like transfer learning and fine-tuning can further improve classification accuracy, especially for challenging or subtle cases.

Another important area for improvement is the integration of the system into real-time applications. Using OpenCV, the project can be extended to operate in real-time, enabling mobile apps or web platforms to analyze skin lesions from live camera feeds. Real-time diagnostic capabilities would not only provide faster results but also make the system more accessible, allowing users to perform self-checks or enabling remote consultations with dermatologists.

Furthermore, developing a user-friendly mobile or web application could improve accessibility. Clinicians and users could upload skin images for analysis, receive instant results, and access further recommendations for treatment or professional consultation. To make this application practical in a clinical environment, the integration of real-time video feeds with automatic lesion tracking and analysis using OpenCV and other libraries will provide timely assistance to healthcare professionals.

In conclusion, automated skin lesion detection holds great promise for revolutionizing healthcare, particularly by increasing the speed, accessibility, and accuracy of skin cancer diagnosis. Ongoing advancements in AI and its integration into real-time applications will further enhance the system's utility and broaden its impact.

6. BIBLIOGRAPHY

6.1 REFERENCES

1. **HAM10000 Dataset**: Tschandl, P., et al. (2018). *The HAM10000 dataset: A large collection of multi-source dermatoscopic images of pigmented lesions*. Available at: https://www.kaggle.com/datasets/kmader/skin-cancer-mnist-ham10000
2. **Nature Report**:
   Esteva, A., Kuprel, B., Novoa, R. A., et al. (2017). *Dermatologist-level classification of skin cancer with deep neural networks*. *Nature*, 542(7639), 115-118. Available at: https://www.nature.com/articles/nature21056+
3. **Official Paper**:
   Litjens, G., Kooi, T., Bejnordi, B. E., et al. (2017). *A survey on deep learning in medical image analysis*. *Medical Image Analysis*, 42, 60-88. Available at: https://www.sciencedirect.com/science/article/abs/pii/S1361841517301135?via%3Dihub
4. **Python Documentation**:
   Python Software Foundation. (2021). *Python documentation*. Available at: https://docs.python.org/3/
5. **TensorFlow Documentation**:
   TensorFlow. (2021). *TensorFlow: An end-to-end open source machine learning platform*. Available at: https://www.tensorflow.org/
6. **Keras Documentation**:
   Keras. (2021). *Keras: Deep learning for humans*. Available at: https://keras.io/