1.INTRODUCTION AND BACKGROUND

A skin lesion is a growth or appearance of the skin that is abnormal concerning the surrounding skin. Primary and secondary skin lesions are the two types of skin lesions. Primary skin lesions are abnormal skin conditions that can develop over time or be present at birth. Secondary skin lesions can develop from primary skin lesions that have been exacerbated or altered. When a mole is scraped until it bleeds, the crust that forms, as a result, develops a secondary skin lesion. Dermatologists propose one of three treatments for afflicted skin, depending on the type of lesion: home care, medicines, or surgery. Regardless of ways innocent they appear; a few sorts of skin lesions may be pretty risky to the patients, since they will indicate the presence of malignancy and require surgical removal. Melanoma is the most dangerous type of skin cancer; as soon as it has spread, it's deadly, however, it is treatable in its early stages. As a result, a precise diagnosis of skin patches is essential to protect patients' growths and ensure that they receive timely treatment.

Machine Learning methods could be used to automate the analysis, resulting in a system and framework in the medical field that would aid in providing contextual relevance, improving clinical reliability, assisting physicians in communicating objectively, reducing errors related to human fatigue, lowering mortality rates, lowering medical costs, and more easily identifying diseases. A machine learning method that can categorize both malignant and benign pigmented skin lesions is a step toward achieving these goals

1.1 Problem statement:

The manual interpretation of skin lesions poses significant challenges in terms of accuracy, timeliness, and accessibility to specialized care. Dermatologists, while possessing expertise, may face difficulties in differentiating between visually similar benign and malignant lesions. Furthermore, the subjectivity in visual assessments and the limitations in accessing dermatological expertise can lead to delayed diagnoses, impacting patient outcomes. Addressing these challenges requires an innovative approach that combines advanced technologies, such as deep learning, to enhance the efficiency and reliability of skin lesion classification.

1.2 Objectives:

1. Develop a deep learning model, specifically a Convolutional Neural Network (CNN), for the automated and accurate classification of skin lesions.

2. Implement transfer learning techniques to expedite the training process and optimize the deep learning

model's performance in accurately identifying and classifying skin lesions.

3. Curate a diverse and comprehensive dataset of skin lesion images, encompassing various dermatological conditions, to train and validate the deep learning model.

4. Enhance the interpretability of the model by incorporating attention mechanisms or Grad-CAM to highlight regions of interest within the images that contribute most to the classification decision.

5.Conduct comparative analyses with existing state-of-the-art models and validate the developed system on an independent test set to demonstrate its effectiveness in real-world scenarios.

1.3 Dataset

The HAM10000 dataset which consists of 10015 images has been used in the proposed work.The HAM10000 dataset is a vast collection of dermoscopic images of pigmented skin lesions which are very common from multiple sources. Datasets with significant class imbalances are fairly common in the medical industry. It is the same with this data set. In the proposed work, it proved to be a significant challenge. The dataset images have a resolution of 600 × 450 pixels and are saved as JPEG formats.

They are manually cropped and cantered around the lesion, as well as modified for visual contrast and color reproduction, at first. Each image and patient had seven features, namely, age of the patient, sex of the patient, lesion id which is a unique identifier for a particular type of lesion, image id which is a unique identification number for an image, dx type for technical validation, Skin lesion's geographical location, and a diagnostic skin lesion category which is a classification of skin lesions that can be used to diagnose a condition.

1.4 Background

Skin lesion classification has emerged as a pivotal field, blending dermatology with artificial intelligence (AI) to address the escalating challenges in diagnosing skin disorders accurately. Traditional diagnostic approaches heavily rely on visual assessment by dermatologists, but the integration of digital imaging and machine learning, especially deep learning, has ushered in a new era. These advancements have led to the creation of automated models proficient in discerning various types of skin lesions, distinguishing between benign and malignant cases. The synergy of medical expertise and computational capabilities holds considerable promise, offering improved efficiency and precision in dermatological diagnoses. This integration has the potential to significantly enhance healthcare outcomes and broaden access to dermatological care.

2. LITERATURE SURVEY

Skin lesion classification, a critical area in medical image analysis, has been the subject of several recent research efforts.In the paper by [1]Solene Bechelli and Jerome Delhommelle et al., the study critically assesses skin tumor classification using machine learning (ML) and deep learning (DL) models. ML algorithms, including logistic regression and k-nearest neighbors, exhibit accuracies below 0.72, improved to 0.75 with ensemble learning. In contrast, DL models, incorporating custom CNNs and pre-trained models (VGG16, Xception, ResNet50), achieve higher accuracies, up to 0.88. Notably, VGG16 demonstrates an F-score and accuracy of 0.88 on a smaller database and metrics of 0.70 and 0.88 on a larger, imbalanced dataset post fine-tuning.

In the paper by [2]Dusa Sai Charan et al., a method is proposed that introduces a multi-model approach, where the output of one model acts as input to the next. The emphasis is on diversifying input strategies, preprocessing techniques, and training methodologies to enhance overall classification accuracy. This approach signifies a comprehensive exploration of model variations to improve the robustness of skin lesion classification systems.

In the work of [3]Peter J. Bevan and Amir Atapour-Abarghouei, the focus shifts towards addressing bias in skin lesion classification, particularly related to skin tone. Leveraging ResNeXt-101 as a feature extractor, the authors incorporate debiasing techniques such as 'Learning Not To Learn' and 'Turning a Blind Eye.' The objective is to achieve a binary classification outcome, distinguishing between benign and malignant cases. This research contributes to the ongoing dialogue about fairness and bias mitigation in medical image analysis.

A different avenue is explored in the research paper by [4]Danilo Barros Mendes and Nilton Correia da Silva, where a deep learning model employing a ResNet-152 architecture is introduced. This model targets the classification of skin lesions in clinical images, including challenging cases such as Malignant Melanoma and Basal Cell Carcinoma. Transfer learning is applied to overcome data scarcity in the medical domain, showcasing the adaptability of deep learning techniques to address real-world challenges. The reported results, with an area under the curve (AUC) of 0.96 for Melanoma and 0.91 for Basal Cell Carcinoma, highlight the promising potential of deep learning in early detection of skin diseases.

Collectively, these studies contribute to the evolving landscape of skin lesion classification by presenting diverse methodologies, from multi-model approaches to bias mitigation techniques and advanced deep learning architectures. The literature survey underscores the ongoing efforts to improve the accuracy,

robustness, and fairness of skin lesion classification systems, ultimately aiming to enhance early detection and diagnosis in the field of dermatology.

In the realm of image analysis, the paper titled [5]"Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization" by Ramprasaath R. Selvaraju et al. stands as a pivotal contribution. While prior studies in skin lesion classification have delved into model architectures, fairness, and clinical adaptability, Grad-CAM introduces a critical facet—interpretability. Published in ICCV 2017, this work presents a method for generating visual explanations from deep neural networks using gradient-based localization. By providing insights into the salient regions influencing model predictions, Grad-CAM addresses the interpretability challenge inherent in complex neural networks. This addition to the literature enriches the ongoing efforts to enhance transparency and trustworthiness in dermatological image analysis, emphasizing the importance of understanding and visualizing the decision-making process of deep learning models for skin lesion classification.

The paper titled [6]"Learning Deep Features for Discriminative Localization" by Bolei Zhou et al. presents a significant stride in advancing the interpretability of deep neural networks. Published in CVPR 2016, the study focuses on learning deep features that facilitate discriminative localization, contributing to the understanding of neural network decisions. By introducing methods to identify and highlight discriminative regions within images, the authors address the interpretability challenge in deep learning models. This work complements existing literature in skin lesion classification, offering valuable insights into the critical task of localizing and understanding relevant features for accurate lesion classification. The methodology and findings contribute to the broader discourse on enhancing transparency and interpretability in medical image analysis, thereby fostering trust in the application of deep learning models for dermatological diagnostics.

3. PROPOSED SYSTEM ANALYSIS

The proposed system revolutionizes disease detection through advanced medical imaging technologies. Integrating data augmentation, deep learning, and transfer learning, our model achieves superior accuracy by training on an extensive dataset. Optimized with cutting-edge techniques, the system features a user-friendly interface for efficient interaction, benefiting both medical professionals and patients.

3.1 Existing System

Skin lesions, arising from diverse causative factors, can manifest on individuals due to abnormal growth in skin tissue, often associated with cancer. This pervasive disease affects over 14.1 million patients worldwide and has been accountable for more than 8.2 million deaths. Addressing the imperative need for accurate diagnostic tools, this study proposes the development of a classification model encompassing 12 lesions, including challenging cases like Malignant Melanoma and Basal Cell Carcinoma. The methodology integrates the utilization of the ResNet-152 architecture, trained on a dataset of 3,797 images, subsequently augmented by a factor of 29 through positional, scale, and lighting transformations. Evaluation of the model involves testing with 956 images, yielding notable results with an area under the curve (AUC) of 0.96 for Melanoma and 0.91 for Basal Cell Carcinoma. The research underscores the significance of leveraging advanced architectures for improved diagnostic accuracy in skin lesion classification, emphasizing the potential for enhanced healthcare outcomes.

3.2 Disadvantages

A notable limitation of the proposed skin lesion classification system is its inherent lack of an explainability factor. Despite its commendable performance in accurately categorizing lesions, the system falls short in providing comprehensive explanations for its decisions. The absence of an inherent explainability mechanism hinders the transparency of the model's decision-making process, which is crucial for gaining insights into the features and factors influencing its classifications. This limitation poses challenges in interpreting and validating the model's outcomes, potentially limiting its practical applicability in clinical settings where interpretability and transparency are paramount. Future enhancements to the system should consider integrating explainability frameworks to foster a deeper understanding of the model's decision rationale and ensure its reliable deployment in real-world medical scenarios.

3.3 Proposed System

Employ advanced data augmentation techniques to systematically expand the dataset, thereby facilitating comprehensive model training on a larger and more diverse set of examples.Leverage sophisticated deep learning architectures to improve disease detection accuracy and unravel intricate features inherent in medical imaging datasets. The application of deep learning models ensures a nuanced understanding of complex patterns within the data.

Integrate transfer learning methodologies by harnessing the pre-trained weights of established models, such as ResNet,MobileNet,InceptionV3. This approach optimizes model performance and facilitates superior learning by leveraging the knowledge acquired during the pre-training phase on large-scale datasets.

Implement state-of-the-art optimization techniques to enhance the model's overall accuracy. Optimization strategies will be employed to fine-tune model parameters and improve convergence, ensuring the model achieves optimal performance on disease detection tasks.

Develop a user-friendly interface tailored for both medical professionals and patients. The interface design prioritizes ease of use, ensuring a seamless experience for doctors and patients alike. This user-centric approach enhances accessibility and facilitates efficient interaction with the disease detection system

3.4 Advantages

The integrated system brings forth a multifaceted approach to disease detection in medical imaging, harnessing advanced data augmentation techniques for comprehensive model training on an expanded and diverse dataset. Employing sophisticated deep learning architectures, including ResNet, MobileNet, and InceptionV3, enhances disease detection accuracy by unraveling intricate features within the imaging data. The incorporation of transfer learning methodologies optimizes performance by leveraging pre-trained weights from established models, expediting adaptation to specific disease detection tasks. State-of-the-art optimization techniques further fine-tune model parameters, ensuring optimal accuracy. Notably, the user-friendly interface caters to both medical professionals and patients, prioritizing accessibility and facilitating efficient interaction with the disease detection system. This integrated approach converges to create a robust and efficient tool for accurate disease detection and classification in medical imaging.

3.5 Requirements

3.5.1 Software Requirements

The software requirements for the integrated disease detection system encompass a versatile set of tools and frameworks tailored to support advanced data processing, deep learning model development, optimization, and user interface design. The key software requirements include:

1. Programming Language:
   - Python: The primary programming language for implementing deep learning algorithms, leveraging popular libraries such as TensorFlow and PyTorch.

2. Data Augmentation Tools:
   - OpenCV: A computer vision library for image processing and augmentation techniques.

3. Pre-trained Models:
   - ResNet, MobileNet, InceptionV3: Pre-trained deep learning models for transfer learning methodologies.

4. Optimization Tools:
   - Optimization Libraries: SciPy, scikit-learn, or similar libraries for optimizing model parameters and improving convergence.

5. User Interface Development:
   - Web Framework: Flask or Django for developing a web-based user interface.
   - Frontend: HTML, CSS, and JavaScript for creating an interactive and user-friendly interface.
   - Visualization Libraries: D3.js or Plotly for data visualization within the interface.

6. Version Control:
   - Git: Version control system for tracking changes in codebase and collaborating with a development team.

7. Integrated Development Environment (IDE):
   - Jupyter Notebooks or Visual Studio Code: Preferred IDEs for interactive development and experimentation with machine learning models.

8. Class Activation Map (CAM) Generation:

- Implementation of CAM techniques using Grad-CAM (Gradient-weighted Class Activation Mapping) Libraries

3.5.2 Hardware Requirements

The hardware requirements for the integrated disease detection system, including the Class Activation Map (CAM) integration, depend on the computational complexity of the deep learning models and the scale of data processing. Here are the key hardware components:

1. CPU:

- Multi-core processors (e.g., Intel Core i5 or higher) for general-purpose computing tasks and data preprocessing.

2. GPU:

- NVIDIA GeForce or Quadro series GPUs (e.g., GTX 1080 Ti, RTX 2080, or higher) for accelerated deep learning model training and inference. CAM generation can benefit significantly from GPU parallelism.

3.RAM:

- Minimum 8 GB RAM for handling large datasets and training deep learning models efficiently.

4. Storage:

- SSD with sufficient capacity (e.g., 500 GB or more) for fast data access, model storage, and efficient handling of augmented datasets.

5. CAM Visualization Requirements:

- High-resolution displays (e.g., Full HD or 4K) for detailed visualization of CAM overlays on medical images.

3.6 Proposed System Architecture:

The proposed architecture consists of ResNet152 along with the integration of CAM. The architectural framework integrates ResNet-152 as its foundational backbone, comprising 152 layers. Commencing with an input layer and an initial convolution layer, it progresses through subsequent layers featuring MaxPool, convolution, and ReLU operations organized into residual blocks. Notably, this architecture is augmented

with the incorporation of Class Activation Maps (CAM), strategically enhancing interpretability by revealing the network's focus during predictions. The final layer consists of Softmax, contributing to precise class predictions. This combined architecture synergizes the depth of ResNet with the interpretability afforded by CAM, establishing a comprehensive framework for robust and insightful image analysis.

It has 2 major components:

3.6.1. ResNet-152 Architecture:

ResNet-152 is employed as the foundational framework, featuring 152 layers. It begins with an input layer, followed by an initial convolution layer, MaxPool, and a sequence of convolutional and ReLU layers structured into residual blocks. ResNet-152 is a deep convolutional neural network architecture introduced in 2016 as part of the ResNet family. Developed by Kaiming He et al., ResNet-152 is known for its remarkable depth, featuring 152 layers. The key innovation lies in its residual learning blocks, which include skip connections to ease the training of deep networks by learning residual functions. This architecture has been particularly successful in image recognition tasks, demonstrating state-of-the-art performance in tasks like image classification, thanks to its ability to effectively handle very deep networks and mitigate issues like the vanishing gradient problem.

The key innovation in ResNet is the introduction of residual learning blocks. These blocks contain skip connections (or shortcuts) that allow the network to learn residual functions, making it easier to train very deep networks. The residual blocks help to mitigate the degradation problem, where the accuracy of a deep neural network saturates and then degrades as the network gets deeper.

In the case of ResNet-152, the architecture is built upon a series of residual blocks. It starts with a traditional convolutional layer, followed by multiple residual blocks organized into different stages. Each stage gradually reduces spatial dimensions while increasing the number of filters. The final global average pooling layer and fully connected layer produce the network's output.

The major characteristics of this architecture are:
- ResNet-152 comprises 152 layers in its architecture.
- The initial layers include the input layer followed by an initial convolution layer.
- A MaxPool layer follows the convolution layer.
- A series of convolutions with ReLU layers are structured into layers, implementing the concept of residual

blocks.

- The final layer is the Softmax layer, facilitating class predictions.

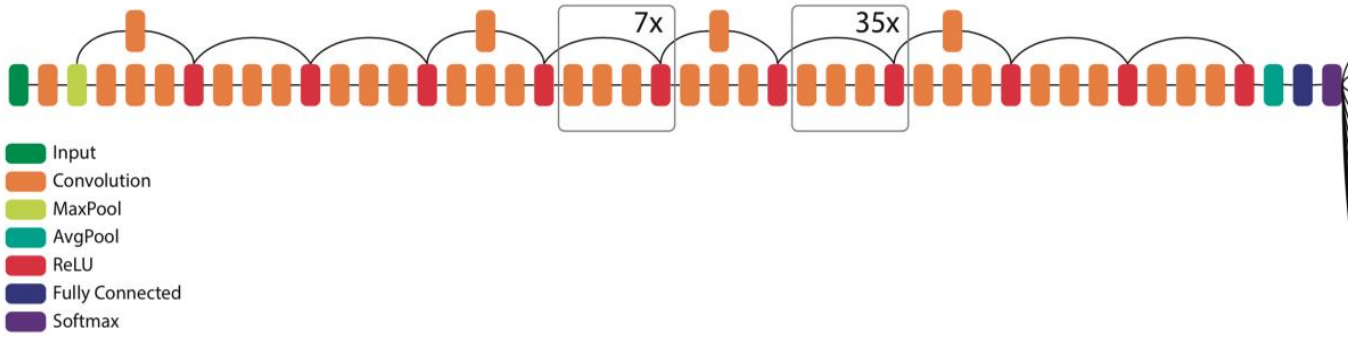

Figure 3.1 ResNet 152 architecture

3.6.2 MobileNet Architecture

MobileNet is a neural network architecture specifically engineered for mobile and edge devices, designed to provide a lightweight yet efficient solution for deploying deep learning models on resource-constrained platforms. Introduced by Google researchers in 2017, the MobileNet architecture addresses the challenge of delivering high-performance computer vision applications on devices with limited computational resources.

The distinctive feature of MobileNet lies in its use of depthwise separable convolutions, a novel approach to convolutional operations. Unlike traditional convolutions, depthwise separable convolutions consist of two separate operations: depthwise convolution, which applies a single convolutional filter to each input channel independently, and pointwise convolution, which uses 1x1 convolutions to combine information across channels. This separation significantly reduces the computational cost compared to standard convolutions, making MobileNet highly efficient without sacrificing accuracy.

MobileNet is organized into multiple layers, each incorporating depthwise separable convolutions, followed by batch normalization and non-linear activation functions. The architecture employs a series of building blocks, including bottlenecks with linear projections, which further enhance computational efficiency. Additionally, MobileNet introduces the concept of 'width multiplier' and 'resolution multiplier,' providing a flexible means to trade off between model size and accuracy, allowing customization based on the specific deployment

requirements.

Over time, the MobileNet architecture has evolved with subsequent versions, such as MobileNetV2 and MobileNetV3, each introducing improvements in terms of performance, accuracy, and adaptability to diverse scenarios. Its impact continues to shape the landscape of model design for resource-constrained environments.

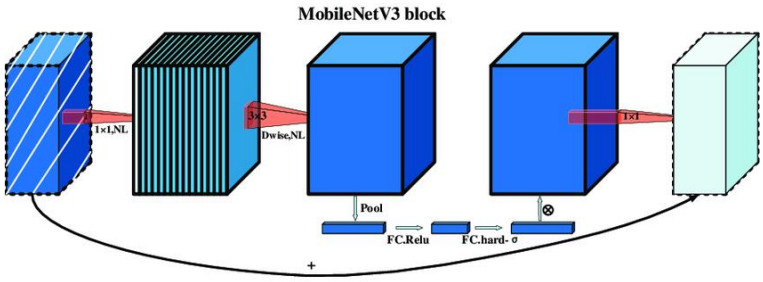

Figure 3.2 MobileNet V3 architecture

3.6.3 InceptionV3 Architecture

InceptionV3, an evolution of the original Inception (GoogLeNet) model, represents a deep convolutional neural network architecture designed for image classification and recognition tasks. Introduced by Google researchers in 2015, InceptionV3 is crafted to strike a balance between computational efficiency and the ability to capture intricate patterns in images. The model begins with a stem network, which performs initial convolutions, pooling, and normalization to extract fundamental features from input images of size 299x299 pixels.

The distinctive feature of InceptionV3 lies in its utilization of Inception modules. These modules consist of parallel branches employing different convolutional filters, including 1x1, 3x3, and 5x5 convolutions, as well as pooling operations. This multi-branch design enables the network to capture features at various scales, enhancing its ability to discern complex patterns. Additionally, reduction blocks are interspersed between Inception modules, incorporating convolutional layers and pooling operations to manage computational complexity and control the network's parameter count.

InceptionV3 further includes auxiliary classifiers at intermediate layers during training. These classifiers aid in mitigating the vanishing gradient problem and provide additional supervision, contributing to better convergence during the training process. The architecture also features fully connected layers towards the end, where global average pooling is applied to reduce spatial dimensions, and the final predictions are generated.

Throughout the architecture, batch normalization and dropout are applied for regularization, preventing overfitting during training. InceptionV3 is recognized for its efficient use of inception modules, allowing it to capture hierarchical features effectively while maintaining a relatively lightweight structure. Its versatility and effectiveness have made it a popular choice in various computer vision applications, demonstrating its proficiency in tasks such as image classification, object detection, and feature extraction.

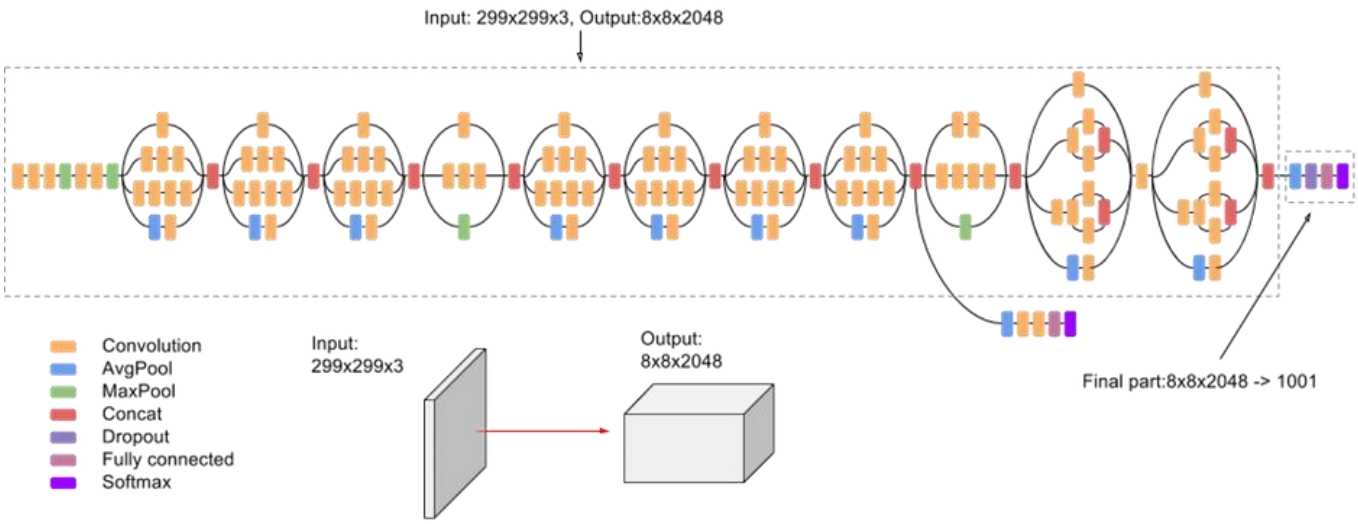

Figure 3.3 Inception V3 architecture

3.6.4 CAM Integration

The architecture is augmented with the addition of Class Activation Maps (CAM). CAM enhances interpretability by revealing the network's focus during predictions, providing valuable insights into the decision-making process for precise image analysis.

Class Activation Maps (CAM) serve as a valuable technique within convolutional neural networks (CNNs) for enhancing model interpretability. Applied typically to the final convolutional layer, CAM generates heatmaps that highlight crucial regions in an input image, offering spatial localization insights. By employing a gradient-based approach, CAM transparently reveals the decision-making process, attributing importance scores to different image regions. Particularly useful in tasks like object localization, CAM aids in both understanding and validating model predictions, providing a detailed visual representation of where the network focuses during classification. Its integration, especially with architectures like ResNet, contributes to a nuanced and insightful analysis of feature importance in image classification tasks.

- The utilization of CAM enhances model interpretability by emphasizing influential regions within images.
- CAM aids in object localization, contributing to a detailed understanding of where the network focuses during predictions. The integration of CAM enhances transparency, providing insights into the decision-making process of the network in image classification tasks.

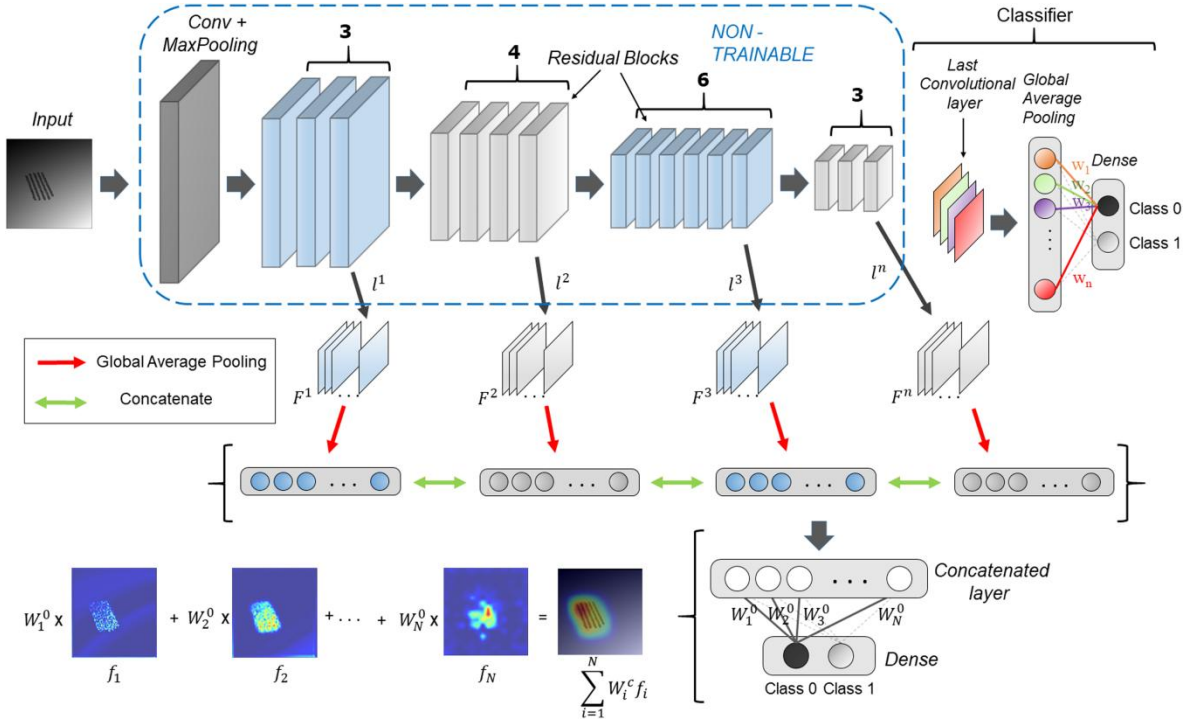

Figure 3.4 Class Activation Maps(CAM)

4.DATASET DESCRIPTION

4.1 Dataset Collection

The HAM10000 dataset is a comprehensive collection of dermatoscopic images of common pigmented skin lesions, primarily designed for research in the field of skin cancer classification and diagnosis. The dataset is named after the Hospital where it was curated.

Dataset Origin:
The HAM10000 dataset was compiled by a team of researchers led by Dr. Noel Codella. The images were sourced from three major dermatology centers: the Medical University of Vienna, the Melanoma Institute Australia, and the University of Queensland. The dataset is notable for its diversity, encompassing various skin lesion types, including benign nevi and malignant melanomas.

Along with HAM10000 datasets from other challenges are also collected and integrated to the existing dataset according to compatability.

4.2 Features description

4.2.1. Image Data:

- The primary feature in the dataset is the dermoscopic image itself, providing a visual representation of pigmented skin lesions. These images are captured using a dermatoscope, allowing for a detailed examination of skin features.

4.2.2. Diagnosis Labels:

- Each image in the dataset is labeled with a corresponding diagnostic category, indicating the type of skin lesion present. The diagnostic labels include categories such as melanocytic nevus, melanoma, basal cell carcinoma, actinic keratosis, benign keratosis, dermatofibroma, and vascular lesion.

4.2.3. Clinical Metadata:

- The dataset includes clinical metadata associated with each image, providing additional context and information related to the patient and the skin lesion. This may include details such as patient age, gender, and other relevant clinical information.

- The metadata includes:

1. lesion_id:
   - A unique identifier assigned to each skin lesion.
   -Enables individual identification and tracking of specific lesions within the dataset.

2. image_id:
   - An identifier associated with the dermoscopic image of the skin lesion.
   - Facilitates the linkage between the metadata and the corresponding dermoscopic images.

3. dx:
   - Represents the diagnostic category or label assigned to the skin lesion (e.g., melanocytic nevus, melanoma, basal cell carcinoma, etc.).
   - Essential for supervised learning tasks, such as training and evaluating machine learning models for skin lesion classification.

4. dx_type:
   - Indicates the type of diagnostic method or source used to determine the lesion category (e.g., 'histo' for histopathology).
   - Offers insights into the reliability and source of diagnostic labels, influencing the interpretation of dataset annotations.

5. age:
   - Represents the age of the patient associated with the skin lesion.
   - Enables age-based analysis and exploration of potential correlations between age and specific lesion types.

6. sex:
   - Specifies the gender of the patient associated with the skin lesion.
   - Facilitates gender-based analysis and may contribute to understanding gender-related patterns in skin lesions.

7. localization:
   - Specifies the anatomical location on the body where the skin lesion is situated.
   - Useful for studying lesion distribution across different body regions and understanding potential

localization-specific characteristics.

4.3 Sample Dataset

The below is an annotated sample of the HAM10000 dataset.

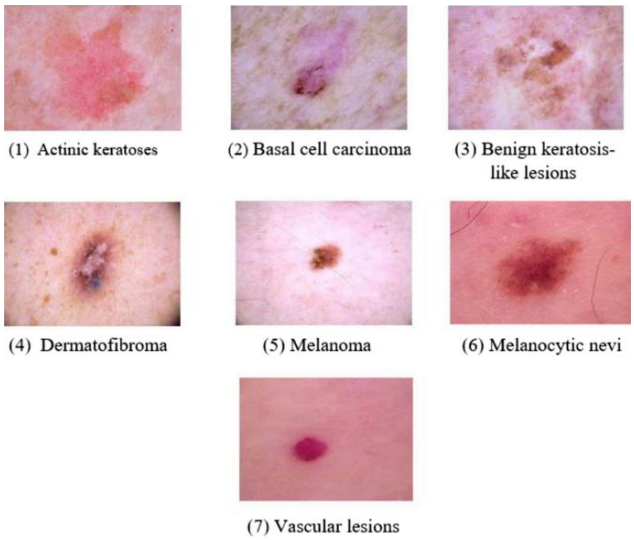

Figure 4.1: Sample Dataset

	A	B	C	D	E	F	G
1	lesion_id	image_id	dx	dx_type	age	sex	localization
2	HAM_0000118	ISIC_0027419	bkl	histo	80	male	scalp
3	HAM_0000118	ISIC_0025030	bkl	histo	80	male	scalp
4	HAM_0002730	ISIC_0026769	bkl	histo	80	male	scalp
5	HAM_0002730	ISIC_0025661	bkl	histo	80	male	scalp
6	HAM_0001466	ISIC_0031633	bkl	histo	75	male	ear
7	HAM_0001466	ISIC_0027850	bkl	histo	75	male	ear
8	HAM_0002761	ISIC_0029176	bkl	histo	60	male	face
9	HAM_0002761	ISIC_0029068	bkl	histo	60	male	face
10	HAM_0005132	ISIC_0025837	bkl	histo	70	female	back
11	HAM_0005132	ISIC_0025209	bkl	histo	70	female	back
12	HAM_0001396	ISIC_0025276	bkl	histo	55	female	trunk
13	HAM_0004234	ISIC_0029396	bkl	histo	85	female	chest
14	HAM_0004234	ISIC_0025984	bkl	histo	85	female	chest
15	HAM_0001949	ISIC_0025767	bkl	histo	70	male	trunk
16	HAM_0001949	ISIC_0032417	bkl	histo	70	male	trunk
17	HAM_0007207	ISIC_0031326	bkl	histo	65	male	back
18	HAM_0001601	ISIC_0025915	bkl	histo	75	male	upper extremity
19	HAM_0001601	ISIC_0031029	bkl	histo	75	male	upper extremity
20	HAM_0007571	ISIC_0029836	bkl	histo	70	male	chest
21	HAM_0007571	ISIC_0032129	bkl	histo	70	male	chest
22	HAM_0006071	ISIC_0032343	bkl	histo	70	female	face
23	HAM_0003301	ISIC_0025033	bkl	histo	60	male	back
24	HAM_0003301	ISIC_0027310	bkl	histo	60	male	back
25	HAM_0004884	ISIC_0032128	bkl	histo	75	male	upper extremity

Figure 4.1: Sample MetaData of the Dataset

5. PROPOSED SYSTEM MODULES

Modules play an important role in the flow of the project. System comprises several crucial modules designed to synergistically enhance disease detection capabilities in medical imaging.Together, these modules form a cohesive and innovative framework poised to significantly advance disease detection in medical diagnostics

List of Modules:

5.1 Data Collection

In the pursuit of advancing medical diagnostics, our project hinges on a meticulous data collection process sourced from various challenges specializing in medical image analysis. Our primary dataset, the HAM10000, has been carefully curated and encompasses a rich repository of 10,000 medical images. Each image within this dataset is accompanied by pertinent labels, crucial for our classification endeavors. This strategic approach to data collection not only underscores the project's commitment to leveraging diverse and high-quality datasets but also positions us at the forefront of innovative solutions in the realm of medical image diagnostics.

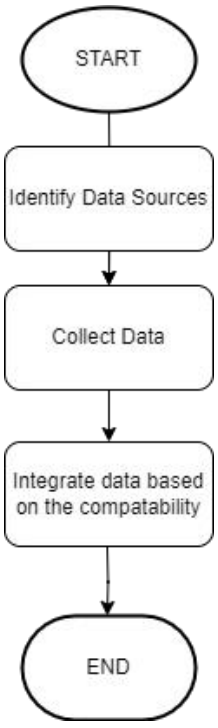

Figure 5.1: FlowChart - Data Collection

5.2 Data preprocessing

In the realm of medical image analysis, effective data preprocessing is paramount to ensure the quality and relevance of the dataset. Initially sourced from various challenges and curated meticulously, our dataset, notably the HAM10000 dataset with 10,000 images, undergoes a series of preprocessing steps to optimize its utility for subsequent tasks.

1. Train-Test Split:

The implementation of a train-test split involves partitioning the dataset into distinct training and testing subsets. This procedural division is essential for evaluating the model's performance on independent data, fostering robustness and generalization.

2. Class Balancing:

The endeavor to balance classes within the dataset is a strategic measure aimed at facilitating model training. By mitigating class imbalances, we aim to prevent the model from developing biases toward a particular class, thereby enhancing its capacity for unbiased and comprehensive learning.

3. Data Augmentation:

Data augmentation represents a pivotal technique involving the augmentation of the dataset through the introduction of artificially generated data derived from existing training samples. This practice serves to diversify the dataset, fortifying the model against overfitting and promoting heightened adaptability.

4. Image Resize, Normalization, and Label Encoding:

This process includes two crucial steps: resizing and normalizing images. Resizing involves adjusting the dimensions of images to ensure a uniform size, which is essential for maintaining consistency in the dataset. Simultaneously, normalizing images involves standardizing pixel values, a crucial step for reducing variations and ensuring that the model processes data consistently.

Alongside these image adjustments, label encoding is applied to convert categorical labels into a numerical format. This transformation streamlines subsequent model training and evaluation processes, enabling the algorithm to comprehend and learn from the labeled data effectively. The combination of resizing, normalizing, and label encoding contributes to a robust preprocessing pipeline, setting the foundation for accurate and efficient machine learning model training.

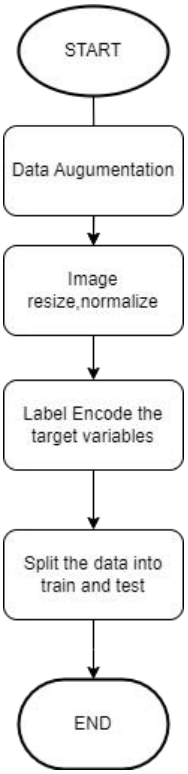

Figure 5.2: FlowChart - Data Preprocessing

5.3 Deep learning Model Selection

The project focuses on crafting a sophisticated deep learning model tailored for image data processing. Leveraging advanced transfer learning, the choice between MobileNet, ResNet, or Inception is made based on specific requirements. Rigorous hyperparameter fine-tuning optimizes model performance for the targeted image dataset. The neural network architecture is systematically configured, with a strategic selection of layers for efficient feature extraction and processing. The approach aims to achieve predefined objectives with precision.

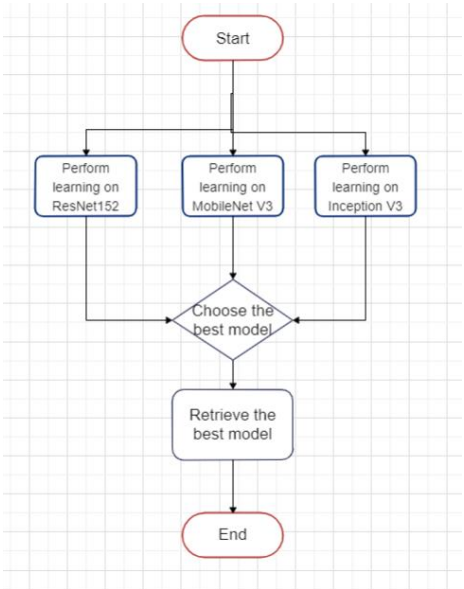

Figure 5.3: FlowChart - Model Selection

5.4 Integration with Explainability Module

Leveraging ResNet-152 architecture with integrated Class Activation Maps (CAM) significantly enhances the interpretability of the neural network's predictions. CAM provides insights into the network's focus during predictions, improving model interpretability. By highlighting influential image regions, CAM facilitates object localization, contributing to a more transparent understanding of the network's decision-making process in image classification tasks. This strategic integration of CAM into ResNet-152 not only bolsters model transparency but also aids in refining the network's performance for nuanced image analyses.

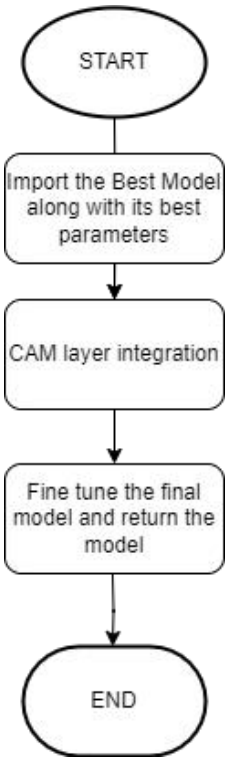

Figure 5.4: FlowChart - Integrate CAM into the Best model

5.5 Performance Evaluation

Evaluation on Challenging Scenarios

To comprehensively assess the performance and robustness of our proposed model, a series of evaluations using diverse and challenging scenarios are conducted. The purpose of these evaluations was to gauge the model's ability to handle real-world conditions and scenarios that may deviate from ideal conditions encountered during training.

1. Low-Quality Images:

- To simulate real-world scenarios where the quality of input images may be suboptimal, we intentionally introduced noise, compression artifacts, and reduced resolution to the images in our evaluation dataset. This test aimed to assess the model's resilience to image degradation commonly encountered in practical applications.

2. Variations in Illumination:

- Illumination variations pose a significant challenge in real-world applications. We evaluated the model's performance under different lighting conditions, ranging from low-light scenarios to high-contrast environments. This assessment aimed to ensure the model's adaptability to varying illumination levels.

Metrics and Analysis:

For each of the challenging scenarios, established evaluation metrics are employed, including but not limited to accuracy, precision, recall, and F1 score. Additionally, qualitative analysis is to be conducted by visually inspecting the model's predictions on a subset of challenging images. This allows to gain insights into the model's behavior in handling specific challenges and identify potential areas for improvement.

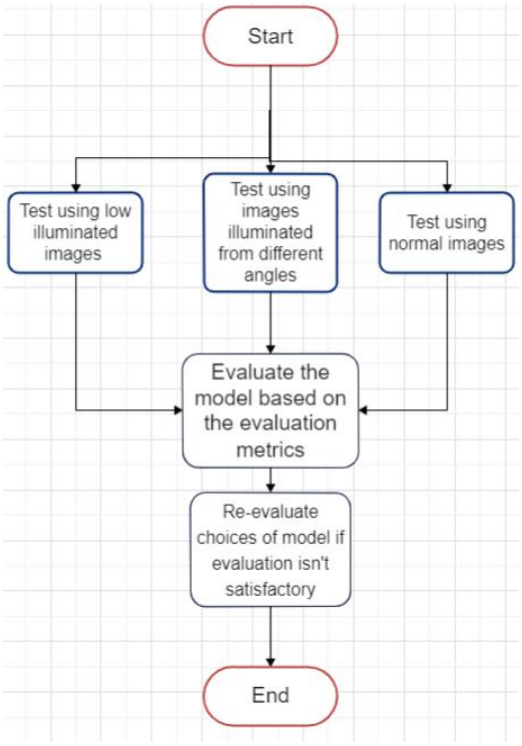

Figure 5.5: FlowChart - Model Evaluation

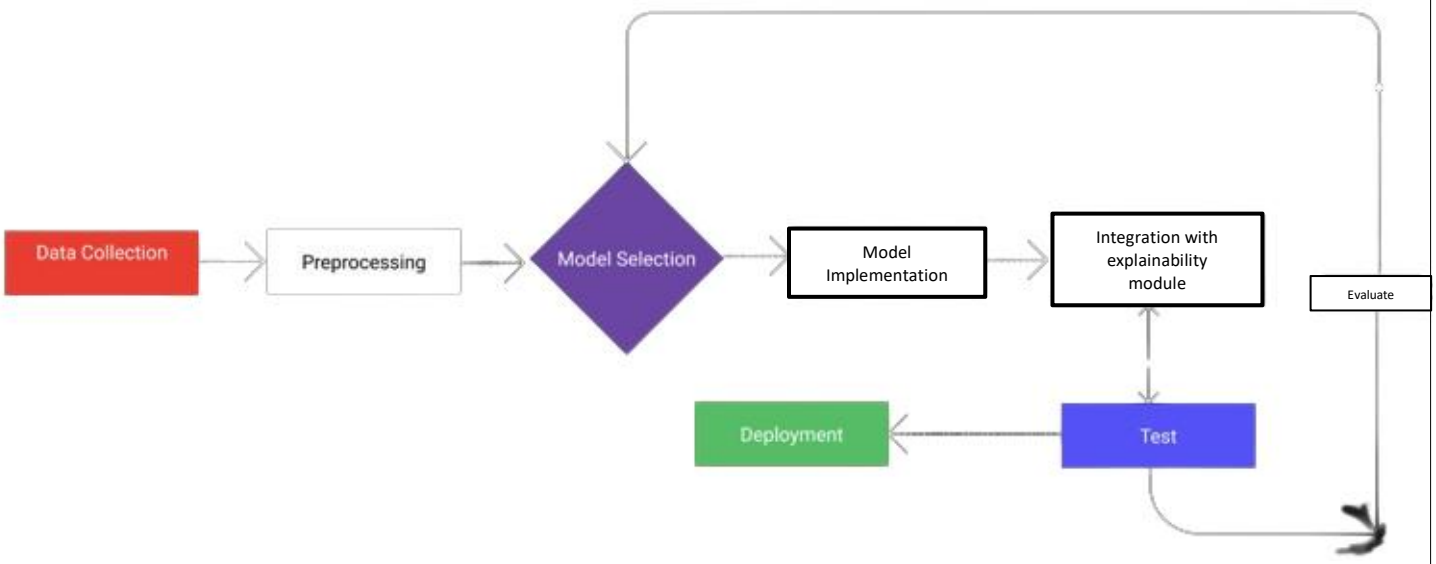

Figure 5.6 Proposed Architecture Modules

6. PARTIAL IMPLEMENTATION WITH ALGORITHMS

6.1 Data extraction:

Obtaining a diverse dataset of skin lesion images is the initial step. This dataset should encompass various types of skin disorders, categorized into benign and malignant cases. The dataset is then split into training and testing sets to facilitate model training and evaluation.

Algorithm - Data Extraction and merging
Input : Kaggle API key
Output: Dataset D

Start
1. Initialize Kaggle API:
2. Set Kaggle API key.
3. Authenticate Kaggle API.
4. Download HAM10000 Dataset:
5. Use Kaggle API to download HAM10000 dataset.
6. Specify the download path.
7. Combine Image Folders:
5. Combine two image folders into one.
9. Specify the paths for the two folders and the combined folder.
10. Store this as Dataset D
return D
End

6.2 Data Augmentation:

To enhance the model's robustness and prevent overfitting, data augmentation techniques such as shear, zoom, and horizontal flip are applied during the training phase. This helps the model generalize better to diverse skin lesion patterns. After augmentation, the data is normalized to ensure consistency and facilitate convergence during training.

To bolster the model's robustness and counteract overfitting, various data augmentation techniques, including

rotate, and horizontal flip, are systematically applied during the training phase. These operations introduce diversity into the dataset, exposing the model to a broader array of skin lesion patterns. Following augmentation, the data undergoes normalization to ensure consistent pixel values, facilitating convergence during training. This dual approach not only enhances the model's adaptability to diverse lesion characteristics but also contributes to its ability to generalize effectively, fostering accurate and resilient skin lesion classification.

Algorithm - Data Augumentation

Input : Dataset D

Output: Augumented Dataset D_a

Start

1. Set Output Directory
2. Specify an output directory path for storing augmented images.
3. Create Output Directory
4. Create the output directory if it does not exist.
5. Set Original Images Directory
6. Specify the path to the original images directory.
7. Set Classes to Augment and Target Sample Count
8. Specify the classes to augment.
9. Set the target sample count for augmentation.
10. Initialize Augmented Entries List
11. Create an empty list to store augmented entries.
12. Iterate Through Metadata
13. For each row in the metadata
14. Check if the class is in the classes to augment.
15. Read the original image.
16. Normalize pixels if the image is not None.

$image_{norm} = image/255.0$

a. Set augment count.
b. For each augmentation:
c. Rotate and flip the image.

d. Rotation

e. If (x,y) are the original pixel coordinates and (x', y') are the rotated coordinates by an Angle θ

$$x' = x.\cos(\theta)-y.\sin(\theta)$$

$$y' = x.\sin(\theta)+y.\cos(\theta)$$

f. Flip

Flipping involves change of sign of the coordinates

$$Flip_x(x,y) = (-x,y)$$

$$Flip_y(x,y) = (x,-y)$$

g. Save the augmented image.

h. Create a new image ID.

$$Image\_id_{new} = Image\_id_{old} + 1$$

i. Create an augmented entry.

17. Create Augmented Metadata DataFrame :

Create a DataFrame from the augmented entries.

18. Merge Metadata:

Merge the original metadata with the augmented metadata into one DataFrame.

19. Store All Images:

Store all the images, including augmented ones, in the specified output directory.

Save it as Dataset D_a

Return D_a

End

6.3 Merge all the datasets

Merging datasets is a crucial step in consolidating diverse sources of information into a cohesive and comprehensive dataset. This process involves combining individual datasets, each representing a unique aspect or source of data, to create a unified and enriched dataset that captures a broader spectrum of information. It ensures a more holistic representation of the underlying phenomena or entities of interest. Merging can be particularly valuable in scenarios where data is collected from multiple sources, such as different experiments, sensors, or diverse data sources, leading to a more nuanced and robust dataset for subsequent analysis. This amalgamation facilitates a more holistic understanding of the domain under consideration and provides a solid foundation for more accurate and generalized model training or analysis.

Algorithm - Merge Datasets

Input:

- List of datasets (datasets_list)

Output:

- Merged dataset (merged_dataset)

Start

1. Initialize an empty dataset to store the merged data
   merged_dataset = []

2. For each dataset Da in the list datasets_list
   a. Append each element in Da to the merged_dataset.

Return merged_dataset

End

6.4 Normalization:

Normalization is a crucial step post-augmentation, ensuring that pixel values across images are standardized. This process helps the model converge faster during training and improves its ability to generalize across different skin lesion images.

In the continuum of data processing post-augmentation, normalization assumes paramount importance. This pivotal step is instrumental in standardizing pixel values across all images, a prerequisite for fostering accelerated convergence during model training. By mitigating variations in pixel intensities, normalization not only expedites the optimization process but also significantly bolsters the model's generalization capabilities. This heightened adaptability across diverse skin lesion images ensures a more resilient and effective model for accurate classification, underlining the critical role that normalization plays in refining the model's performance post-augmentation.

Algorithm - Normalize and Resize Images

Input:

- List of image paths (image_paths)
- Dataset D_a
- Target size for resizing (target_size)

Output:

- List of normalized and resized images (normalized_resized_images)

Start

1. Initialize an empty list to store normalized and resized images:
   Normalized_resized_images = []

2. For each image path in the list image_paths:
   a. Read the image from the file path.
   b. Convert the image to a format suitable for processing (e.g., RGB).
   c. Resize the image to the target size:
      resized_image = cv2.resize(normalized_image, target_size)
   d. Convert them into grayscale images:
      $I=0.299\times R+0.587\times G+0.114\times B$
   e. Convert them into numpy arrays:
      If I is the intensity of the pixels at (x,y) then
         Img_array[x,y] = I
   d. Normalize the pixel values of the image:
      - If the pixel values are in the range [0, 255], apply normalization:
        normalized_image = image/255.0
   e. Append the normalized and resized image to the list normalized_resized_images.

Return normalized_resized_images

End

6.5 Choose the best model:

Examining three prevalent deep learning architectures—ResNet, MobileNet, and Inception—reveals distinctive features crucial for optimal model selection. ResNet's deep and residual learning design proves advantageous for tasks demanding high accuracy and intricate feature learning. However, its computational demands may pose challenges in resource-constrained environments. MobileNet, tailored for edge and mobile applications, strikes a balance between efficiency and accuracy, making it suitable for real-time scenarios. Inception, with its diverse inception modules, achieves equilibrium between accuracy and computational efficiency, though its implementation complexity and model size warrant careful consideration. The nuanced decision-making process involves evaluating task-specific needs, dataset characteristics, and deployment constraints, ensuring an informed selection aligns with the dynamic landscape of deep learning applications.

Algortihm -Model Comparison:

Input:
- Dataset (D_a) with labeled samples for training and evaluation.
- Models: ResNet-152, MobileNet, InceptionV3.

Output:
- Best performing model($model_{best}$)

Begin
1. Data Preprocessing:
Split the dataset into training and evaluation sets.

2. Model Training:
Train each model (ResNet-152, MobileNet, InceptionV3) on the training set using appropriate hyperparameters.
Utilize a common evaluation metric (e.g., accuracy, F1-score) to assess the models' performance during training.Utilizing accuracy

$$\text{Accuracy} = \frac{\text{Total number of true positives and true negatives}}{\text{Total number of samples}}$$

Utilizing F1 Score

$$\text{F1 Score} = \frac{2*\text{number of true positives}}{2*\text{number of true positives} + \text{number of false positives} + \text{number of false negatives}}$$

3. Model Evaluation:

Evaluate the trained models on the evaluation set using the chosen metrics.

Calculate the performance metric for each model.

4. Model Selection:

Identify the model with the highest performance based on the evaluation metric.

If multiple metrics are considered, use a weighted combination or prioritize the most critical metric based on the application.

5. Result Output:

Output the selected model as the best-performing model($model_{best}$) for the given dataset and evaluation metric.

return $model_{best}$

End

6.6 Add explainability factor to the best model:

This algorithm, designed to enhance interpretability, adds a Class Activation Map (CAM) explainability module to the best-performing model. It begins by loading the saved best model and the image designated for CAM visualization. The image undergoes preprocessing to align with the model's input requirements. The algorithm then identifies the last convolutional layer and classifier layer indices in the model. By leveraging TensorFlow's GradientTape, it calculates gradients of the predicted class concerning the output feature map. Subsequently, the algorithm computes the CAM through global average pooling, normalizes it to values between 0 and 1, and creates a heatmap using these values. The final step involves overlaying this heatmap onto the original image, producing the CAM visualization. The outcome, consisting of the original image and the overlaid CAM, is then displayed, providing valuable insights into the model's decision-making process.

Algorithm - Add Explainability module(CAM) to Best Model:

Input:

- Best performing model ($model_{best}$)

- Image for CAM visualization

Output:

- CAM visualization overlaid on the original image

Begin:

1. Load Best Model:
   Load the saved best-performing model($model_{best}$).

2. Load Image for CAM Visualization:
   Load the image on which CAM will be visualized.

3. Preprocess Image:
   Preprocess the image to match the input requirements of ($model_{best}$).

4. Get Relevant Layers:
   Identify the last convolutional layer and classifier layer indices in ($model_{best}$).

5. Calculate Gradients:
   Using TensorFlow GradientTape, calculate gradients of the predicted class with respect to the output feature map.

6. Obtain CAM:
   Perform global average pooling to obtain the CAM.

7. Normalize CAM:
   Normalize the CAM to values between 0 and 1.

8. Heatmap Visualization:
   Create a heatmap using the CAM values.

9. Overlay CAM on Original Image:
   Superimpose the heatmap on the original image to create the CAM visualization.

10. Display Result:
    - Display the original image and the CAM visualization.

Return visualization
End

6.7 Deployment:

Deploying a machine learning model involves a multi-faceted approach encompassing both technical deployment and user interface/user experience (UI/UX) design considerations. The initial step is to select a deployment platform, whether cloud-based or on-premises, ensuring compatibility with the model's infrastructure requirements. Once deployed, creating an effective UI/UX is crucial for user interaction. The UI design should be intuitive, offering a seamless experience for users to interact with the model. This involves designing clear and user-friendly interfaces that accommodate input parameters, showcase model outputs, and provide feedback. A well-crafted UX ensures a smooth and engaging user journey, facilitating users in understanding the model's functionality and making informed decisions. It's imperative to iteratively test and refine the UI/UX design, taking user feedback into account to enhance usability and overall satisfaction. The combination of robust deployment strategies and thoughtful UI/UX design contributes to a successful integration of the model into real-world applications. Once deployed, the model performs its prediction.

Algorithm - Disease Prediction

Input
- image of the skin lesion
- Deployed model($model_{deployed}$)

Output
- Prediction of the disease along with the visualization of where it is

1. Load Deployed Model:
   Load the deployed mode($model_{deployed}$)l.

2. Load Image for Prediction and CAM Visualization:

Load the image for which predictions and CAM will be visualized.

3. Preprocess Image:

Preprocess the image to match the input requirements of

img_array = Preprocess(img)

4. Make Prediction:

Obtain model predictions for the input image:

predictions = $model_{deployed}$(img_array)

5. Obtain CAM:

Obtain CAM visualization

6. Display Result:

Display the original image, predictions, and the CAM visualization.

Return results

End

7. EXTENSION PLAN

The primary goal of project extension is to enhance the functionalities and introduce new features. This extension plan encompasses various aspects, including model selection, integration of CAM, performance optimization, improvements in user interface/experience, and thorough testing. Additionally, documentation and will be prioritized to ensure a smooth rollout of the extended features.

7.1 Model Selection:

Choosing an appropriate model that caters to the needs of the problem in hand the different models which are prospects are RESNET, MobileNet,Inception.

7.2 Integration of CAM

Once the model is created the class activation maps(CAM) are to be integrated in the architecture of the model.

7.3 Performance Optimization

Apply optimization techniques to yield better results all while not compromising on the user experience.

7.4 User interface

Create a user interface which is handy and easy to use.It has to provide the explanation to the output generated.

8. REFERENCES

[1] Bechelli, S., & Delhommelle. Machine Learning and Deep Learning Algorithms for Skin Cancer Classification from Dermoscopic Images.

[2] Dusa Sai Charan, Hemanth Nadipineni, Subin Sahayam, Umarani Jayaraman. Method to Classify Skin Lesions Using Dermoscopic Images.

[3] Peter J. Bevan and Amir Atapour-Abarghouei. Skin Tone Detection and Debiasing for Skin Lesion Classification

[4] Danilo Barros Mendes and Nilton Correia da Silva.Skin Lesions Classification Using Convolutional Neural Networks in Clinical Images

[5] Ramprasaath R. Selvaraju1, Michael Cogswell ,Abhishek Das, Ramakrishna Vedantam, Devi Parikh, Dhruv Batra Grad-CAM:Visual Explanations from Deep Networks via Gradient-based Localization

[6]Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva, Antonio Torralba.Learning Deep Features for Discriminative Localization

9. BIBLIOGRAPHY

- CPU - Central Processing Unit
- RAM - Random access memory
- SSD - Solid State Drive
- GPU - Graphics Processing Unit
- CAM - Class Activation Maps
- ReLU - Rectified Linear Unit