Ain Shams University

Faculty of Computer & Information Sciences

Information Systems Department

Skin disease detection

July 2021

Ain Shams University
Faculty of Computer & Information Sciences
Information Systems Department

Skin disease detection

By

Ahmed Khaled mowaed	**Information Systems**
Hazem Abd Elnasser	**Information Systems**
Mahmoud Saeed Mohamed	**Information Systems**
Mohamed Adel Fathi	**Information Systems**
Mohamed Abd Elhafez	**Information Systems**

Under Supervision of :

Prof , Dr . Rasha Ismail

Information Systems Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

TA. Eman Bahget

Information Systems Department,
Faculty of Computer and Information Sciences,
Ain Shams University.

Contents

Acknowledgment

All praise and thanks to ALLAH, who provided us the ability to complete this work. we hope you accept this work from us.

We are grateful for our parents and our family who were always providing us with support throughout our years of study.

We hope one day we can give back to them. We also offer our sincerest gratitude to our supervisors, **Prof.Dr. Rasha Ismail** and **T.A Eman Bahget** who have supported us throughout our thesis, with their patience, knowledge, and experience.

Finally, We would like to thank our friends for their support and encouragement.

Abstract

Skin diseases are more common than other diseases. Skin diseases may be caused by fungal infection, bacteria, allergy, or viruses, etc. The advancement of lasers and Photonics based medical technology has made it possible to diagnose the skin diseases much more quickly and accurately. But the cost of such diagnosis is still very expensive and limited. So, image processing techniques help us to build applicatons for dermatology at an initial stage. The extraction of features plays a key role in helping to classify skin diseases.deep learning has a role in the detection of skin diseases in a variety of techniques. This work contributes in the research of skin disease detection. We proposed a deep learning-based method to detect skin diseases. This method takes the digital image of the patient effected skin area, then use image analysis to identify the type of disease. Our proposed approach is simple, fast and does not require expensive equipment other than a smart phone with a camera .The approach works on an image sent by the patient then the system extract features using pretrained or customized deep learning techniques. Features are being classified using convolutional neural network . Finally, the results are shown to the user, the type of the disease . Our app doesn't eliminate the necessity for doctors as their roles are indispensable at the moment.

The system successfully detects various different types of skin diseases with an accuracy rate of 87%.

1- Introduction

1.1 Motivation

The last couple of years it has been harder to move around freely due to the coronavirus epidemic, with that in mind we have to find ways to go on with our lives with out putting ourself's through the risk of getting inffected. There is no doubt that going to a doctor to get diagnosed is very important but as said before due to the coronavirus epidemic that situation has become very risky on the patient and on the doctor. we as a team decided to develop skin disease detection application as to provide a solution to one of many problems occurred recently in our world . In general, skin diseases are chronic, infectious and sometimes may develop into skin cancer. Therefore, skin diseases must be diagnosed early to reduce their development and spread. A skin disease may change texture or color of the skin. The diagnosis and treatment of a skin disease takes long time and causes financialy costly to the patient.

In general, most of the common people do not know the type and stage of a skin disease. Some of the skin diseases show symptoms several months later, causing the disease to develop and worsen further.This is due to the lack of medical knowledge in the public. Sometimes.
The advancement of lasers and photonics based medical technology has made it possible to diagnose the skin diseases much more quickly and accurately. But the cost of such diagnosis is still limited and very expensive and pateints still need to go to the hospital to get diagnosed. Therefore, we propose an image processing-based approach to diagnose the skin diseases.This method takes the digital image of disease effect skin area then use image analysis to identify the type of disease. Our proposed approach is simple, fast and does not require expensive equipment's other than smart phone with a camera.

1.2 Problem Definition

Now day's skin diseases become more common problem in human life. Most of these diseases are dangerous and harmful, particularly if not treated at it's initial stages. People do not treat skin diseases seriously. Sometimes, most of the people treat these infections of the skin using their own household methods. However, if these household treatments are not suitable for that particular skin problem then it would affect the skin. Also they may not be aware of severe problem of skin diseases. Skin diseases have tendency to pass from one person to another person easily.Hence it is very important to control it at earlier stage to prevent it from spreading in people. People usually don't want to spend money and time to see the doctors. There are many families who cannot afford this expensive examinations also due to the coronavirus epidemic going to see the doctor to get dignosed has a high risk of infection on the patient and the doctor. It can also become very hard to identify sometimes as many disease have the same symptoms.In order to get the best conclusion, different test needs to be done , so it become an important thing to treat these skin diseases properly at the earlier stages itself to prevent serious damage to skin.Our app allows users to detect the skin diseases to provide treatments or advice to patient .

1.3 Objective

We aim to create a Mobile Application which will be helping in the classification of the skin diseases based on image of affected area, so that the treatment can be initiated early.

Help with solving misdiagnosing problems The application will have easy user friendly GUI.

Users would be able to use phone's camera to take a photo of their skin or upload an already taken photo from there device The application would support many skin disease kinds.pateint would be able to use it from home with putting himself through the risk of getting iffected.

1.4 Time Plan

1. **Survey on the topic:** do a detailed research about skin disease problems, types diagnoses and how to help to prevent it .

2. **System design:** brainstorming of the feature that our project will include and used tools.

3. **Collecting data:** Collecting the required dataset needed for training the model.

4. **Implementation:** Implement our Convolution Neural Network and integrate it with Android application.

5. **Testing:** Implement android application to test our model performance.

6. **Documentation**

And this show our time plan:

Task ID	Task Description	Duration	Start Date	End Date
1	Survey on the topic	10	Nov 01, 2020	Nov 11, 2020
2	System design	20	Nov 12, 2020	Dec 02, 2020
3	Collecting data	20	Feb 10, 2021	Mar 06, 2021
4	Implementation	40	Mar 09, 2021	Jan 05, 2021
5	Testing	20	Apr 20, 2021	May 15, 2021
6	Documentation	15	May 18, 2021	Jun 05, 2021

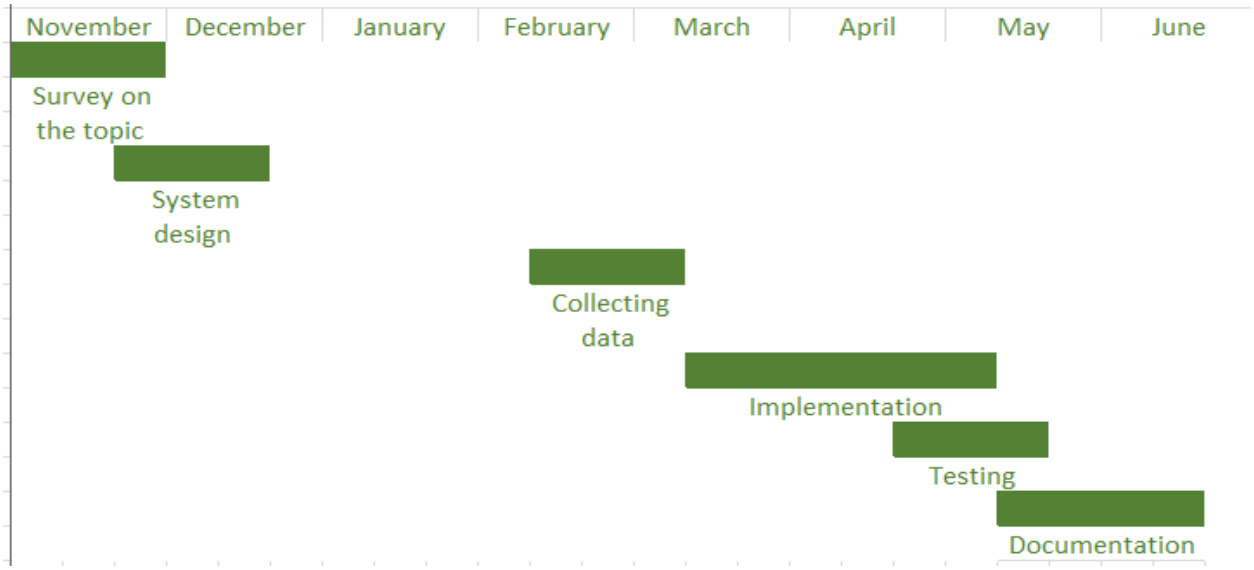


Figure 1 is showing our time plan

2- Analysis and Design

2.1 System Overview

2.1.1 System Architecture

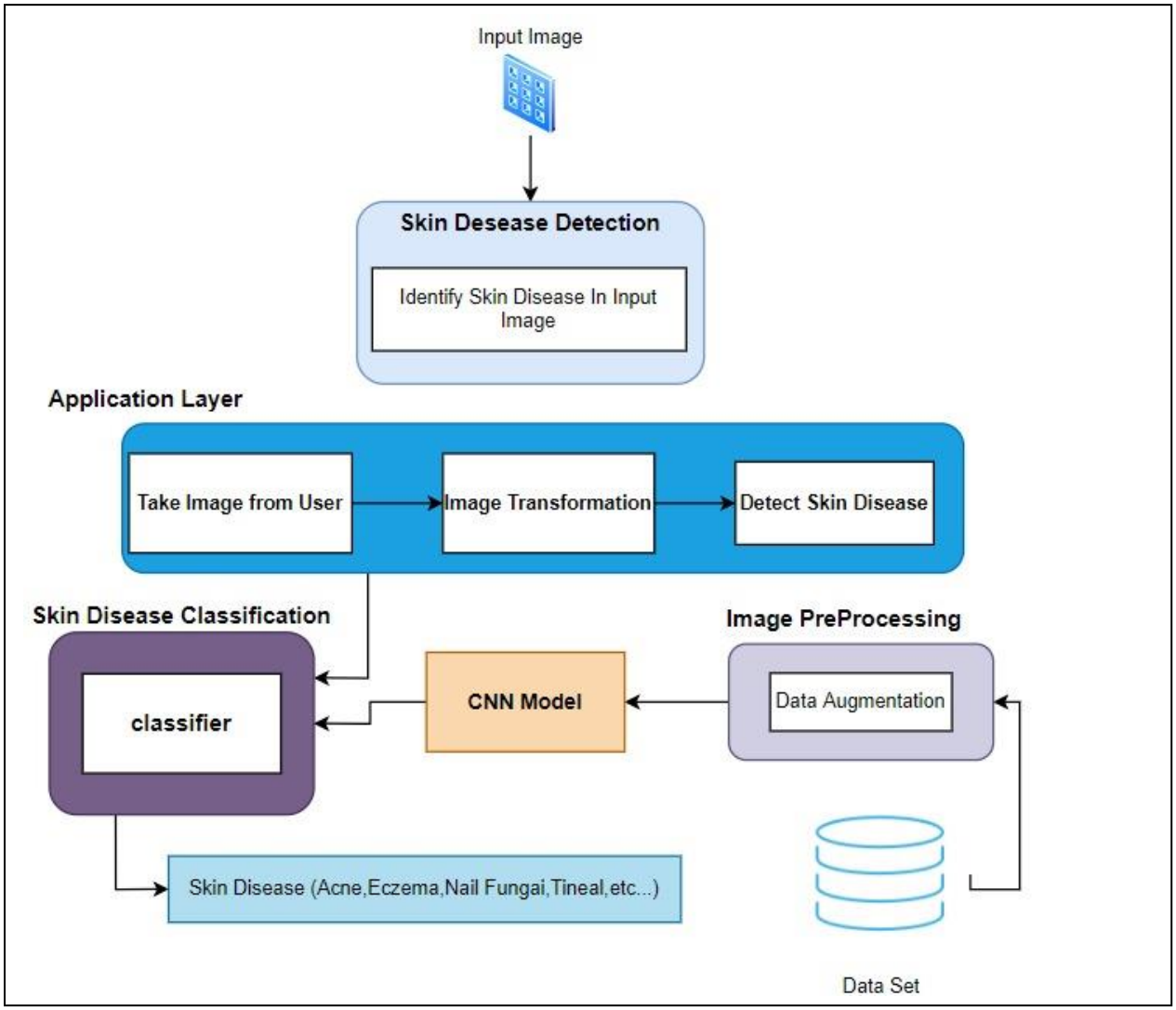
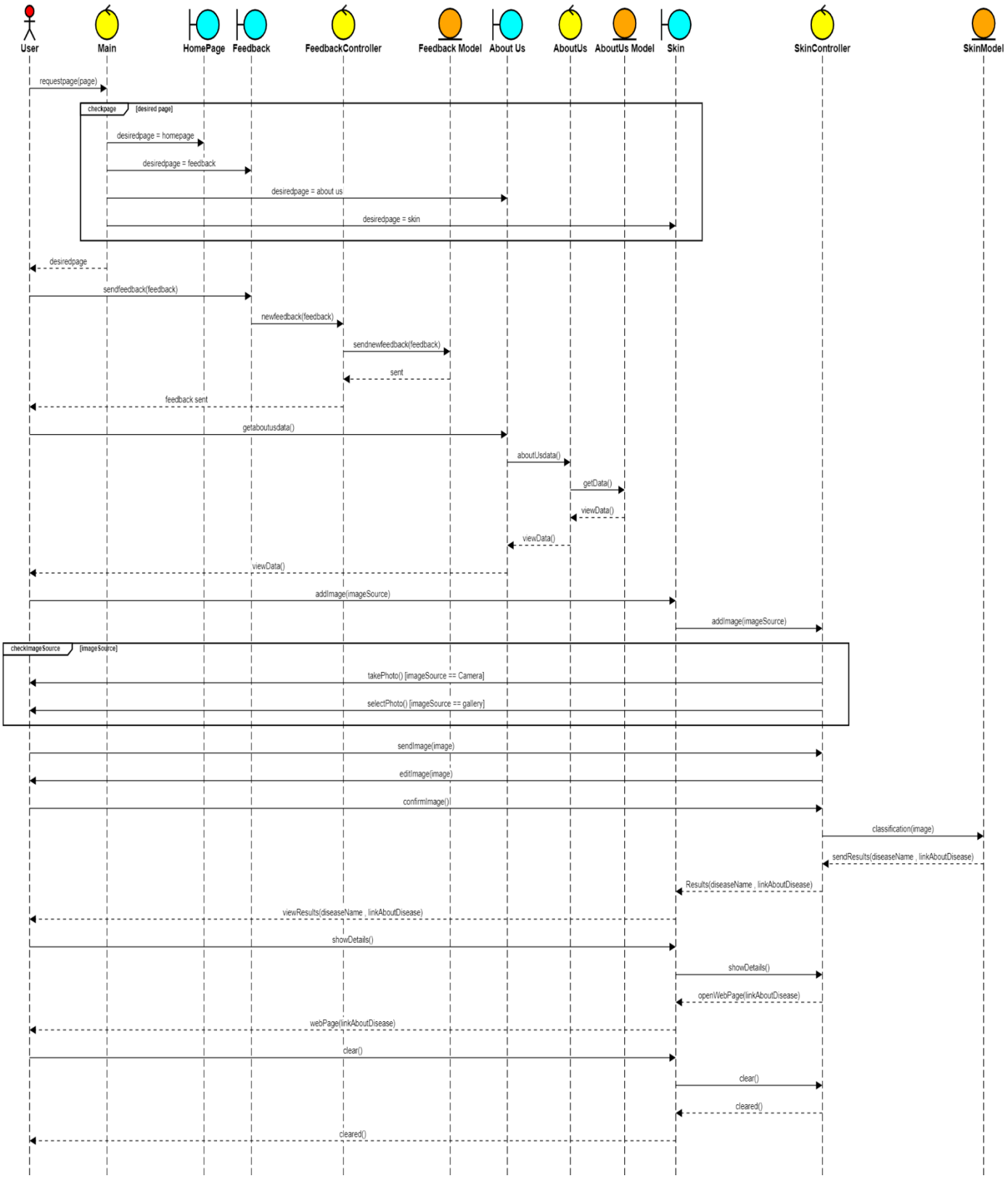


Figure 2 is showing our system architecture

2.1.2 Sequence diagram



First, We take an image of skin from users by using an android application as input.

In the Application layer after taking the image, the user could Zoom and Rotate it to make it clear to detect the diseases, and the CNN algorithm could make image processing to our image then classify the diseases from our CNN Training.

Workflow:

1) Obtain the image of Skin using camera.
2) Sending image to Model
3) Skin classification a using CNN algorithm.
4) Send back classification results as text to user.

2.1.3 Functional Requirements

1) Take pictures using the device's camera.
2) Obtaining images by viewing the phone's gallery and using a specific image for diagnosis
3) Image editing :
    1) Zoom image .
    2) Rotate image .
4) Classify skin diseases .
5) Get Disease Identefication.
6) User Can Send A FeedBack To Our E-Mail.
7) Can See in About Us Form The Team Members and ID.
8) The user, after Sending his Image To The Model, if he has a disease, can Get More Details From Authenticated Webpages for type of this disease, its characteristics and more details.

2.1.4 Nonfunctional Requirements

- **Availability**: The system is available all the times for all the users who want to benefit from it.

- **Maintainability**: The system should be maintainable in such a manner that if any new requirement occurs then it should be easily incorporated in an individual module.

- **Usability**: The system is easy to use. the user is able to learn, prepare inputs, and interpret outputs through the interaction with the system.

- **Reusability**: The system should be reusable in order to create other software from its predefined components. It should help increase productivity, enable interoperability, and reduce maintenance cost.

- **Performance**: The system is interactive, the delay time is considered minimum, and response time is proportional to the size of the dataset.
- **Reliability**: The system performs the specified functions in a consistent-way,without failure.

2.1.5 System Users

A. Intended Users:

Under these circumstances of the Corona pandemic, most people are afraid of gatherings or visiting a doctor, especially dermatologists that need close distance and even touching The Patient to Diagnose the disease.

So our idea came to help users to discover their disease on their own through a program that works on just taking a picture of the skin And preview it and output the result of this preview through a text to the user.

B. User Characteristics:

Anyone can deal with the program, its advantage is that it is simple to deal with all groups: they only need to take a picture.

2.2 System Analysis & Design

1.2.2 Use Case Diagram

This is our use case as shown in the following figure:

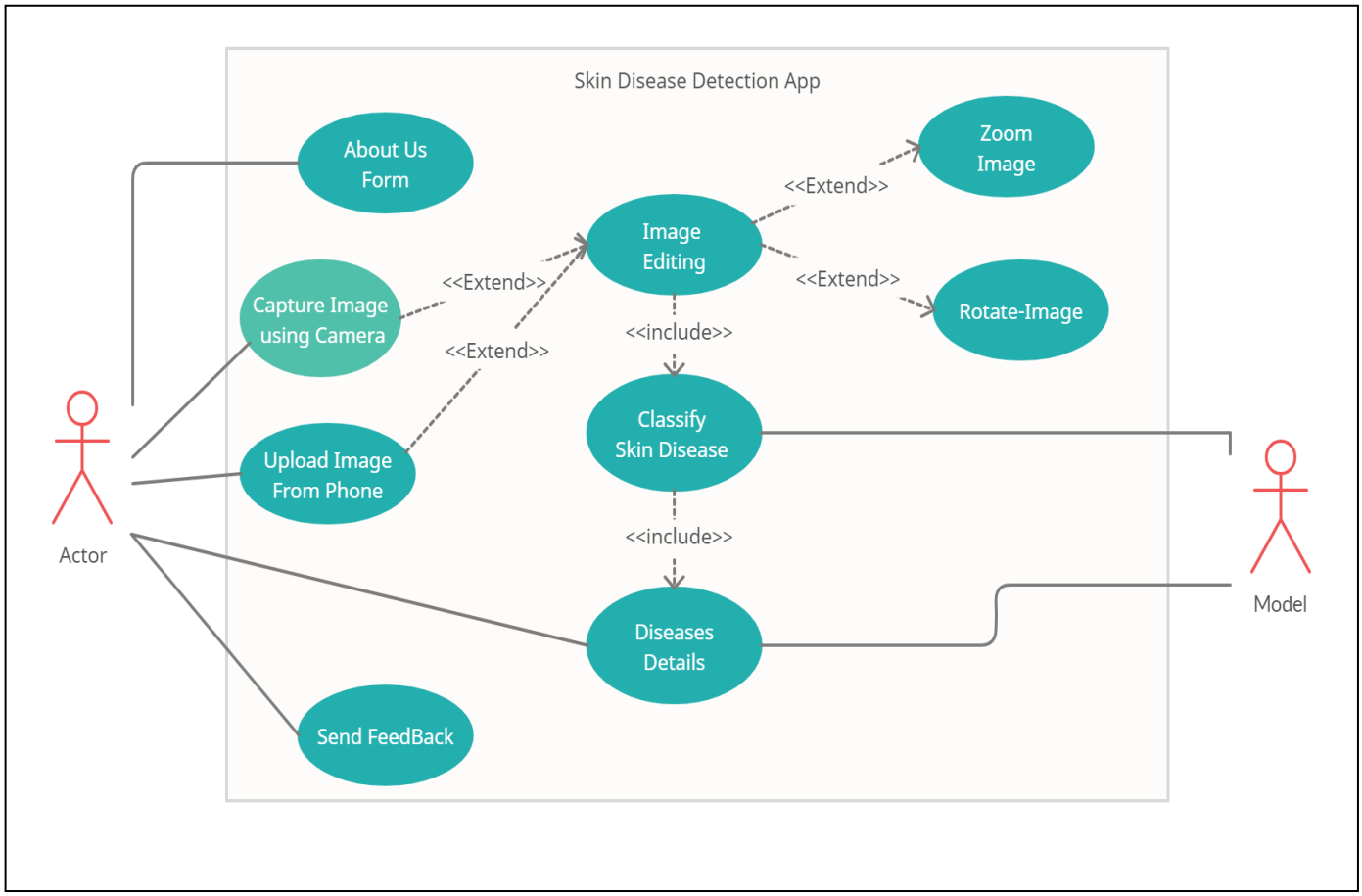


Figure 3 is showing use case diagram

Functions:

- **About Us Form** : The User Can See The Team Members as More Details.
- **Capture Image Using Camera**: the user can Capture photo by using phone's Camera.
- **Upload Image From Phone**: The User can upload Image Form His Gallery.
- **Editing Image** : the User Can make Zoom and Rotate to make The Image More Clear.
- **Classify Skin Disease**: the Model Can Classify The Type of The Disease Form The Image Sent.
- **Disease Details**:view Details About User's Disease.
- The user Can Send A **FeedBack** To us.

2.2.2 Activity Diagram

This is our activity diagram as shown in the following figure:

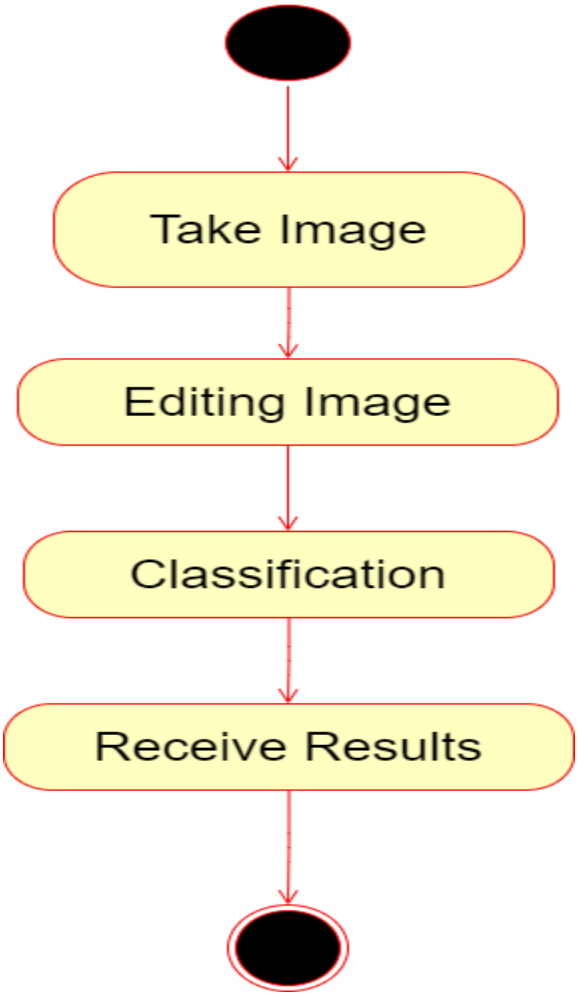


Figure 4 is showing activity diagram

Activities:

1) Take image: receive image from user
2) Editing Image: do Editing functions on received image(Zoom & Rotate).
3) Classification: classify Skin disease Images
4) Receive result: send classification result to user in text

3- Project overview

3.1 Related works

Medgic - Scan, Analyze and Detect Skin Problems

Magic's proprietary Artificial Intelligence (AI) algorithms allow scanning, detection and analysis* of your skin on mobile phones in THREE SIMPLE STEPS:

1) Use Medgic AI camera to take a photo of your skin condition or disease

2) Medgic AI brain will analyse.. No humans involved!

3) Medgic will tell you the results*, along with some friendly advices

FEATURES:

- Camera to capture skin photographs and images

- To detect, appraise and check for dermatological disease or conditions

- To analyze skin health and test for any dryness or inflammation

- To scan, evaluate and recognize any skin problems

- To suggest friendly advices for general good health, fitness and healthy lifestyle.

Smart skin scanner:

- Android application that uses machine learning to find skin disorders that are similar to your's
- The application takes the picture from camera does the classification then it shows top 10 images similar to your case

- Applications mentioned previously are the only applications on android phone that uses machine and deep learning techinques to diagnose diseases related to human skin. But there are some issues with these applications that we tried to solve such as :

- The first application can classify many skin diseases and give brief information about the patient's disease, medgic relies on live footage from the user's camera but the application doesn't give the user the ability to capture and image or use an image which has been already taken from the user's gallery. so we decided to integrate such feature in our mobile android application where the user can select an image which has been already taken and send it to the model and give the user the result back.

- The second application is smart skin scanner, this application also gives the user the ability to take an image and send it to the model but the model returns back top 10 similar diseases to the classified one from the user. Also the application is really far away from being user friendly. So in conclusion such an application will be really difficult for the user to deal with and it might confuse the patient to decide his problem . as a result of what we have stated before we decided to build our mobile application to be user friendly easy to use for all types of users whether new android users or the old ones.

3.2. Background

A. Proposed system:

- Obtain the image of the diseased part of the skin using the camera.
- Use Image Preprocessing algorithms to change the taken image.
- disease classification using CNN algorithm.
- Using the transfer learning concept
- Integrate the CNN model in Android application that can be used by end user.

And the following figure shows an overview of our system:

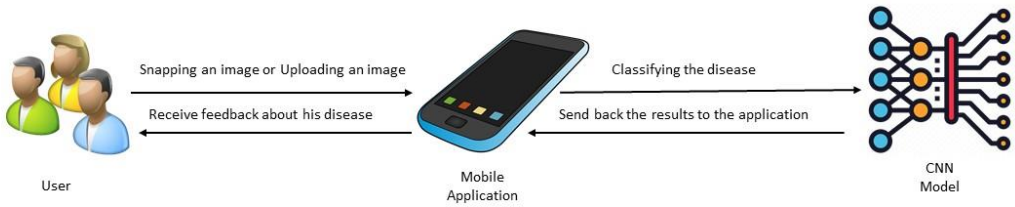


Figure 5 is showing an overview of our system

B. Description of the proposed system:

The proposed system aims to develop a system that can be easily applied to several different diseases and has good efficiency at a high speed.

obtaining the Image:

An image can be obtained using a camera or selecting the image from the user's gallery after requesting a permission from the user.

Preprocessing operations:

- Data quality is essential to get good results
- The goal is to improve the quality of data to ensure that the measurements provided are:
  - Accurate
  - Precise
  - Complete
  - Correct
  - Consistent

Figure 6 is showing example of dataset

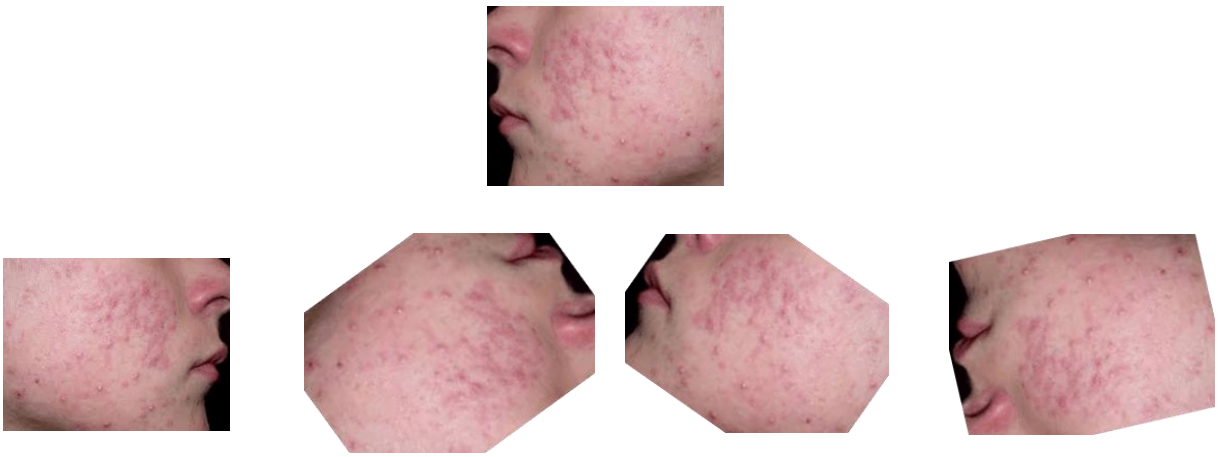
- **Data Preprocessing**

  As mention previously that data quality is essential to get good results and to Train the model efficiently. Getting large amout of data is mandatory so the model can be trained and give high accurate results .

  There are some alternative techniques to getting a large dataset to improve the quality of the data incase that the acquired dataset is relatively small.

  Data Augmentation is considerd a solution to the size of the dataset problem.

  By applying different transformations on the available dataset to synthesize new data. This approach of synthesizing new data from the available data is referred to as ‘Data Augmentation’.



This Figure shows some examples of images augmentations

Each individual image in the dataset is replicated many times with different transformation which leads to dataset Enlargement. By creating many copies of each individual image in the dataset the dataset can be increased. The replicated images isn’t the same when they are copied but each images is changed either by flipping the image, zooming , mirroring and much more transformations can be done to the image to enlrage the dataset.

Using Convolutional neural networks (CNNs) for Image Recognition:

- Convolutional neural networks (CNNs) are widely used in a pattern- and image-recognition problems as they have several advantages compared to other techniques.
  First let's show what is CNN and why we used it.
  That's what we will explain in the next part.
  And this figure shows CNN layers.

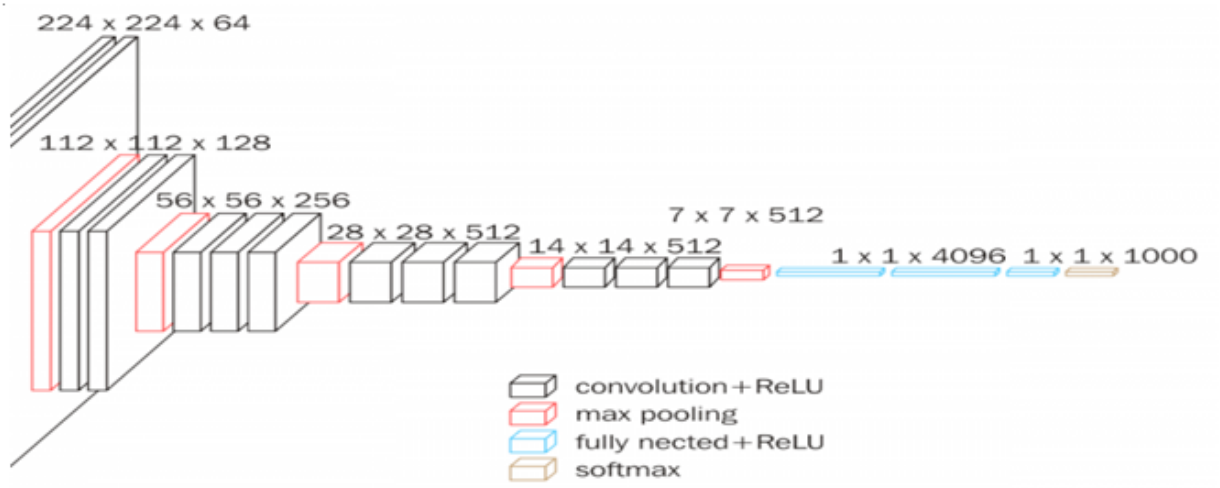
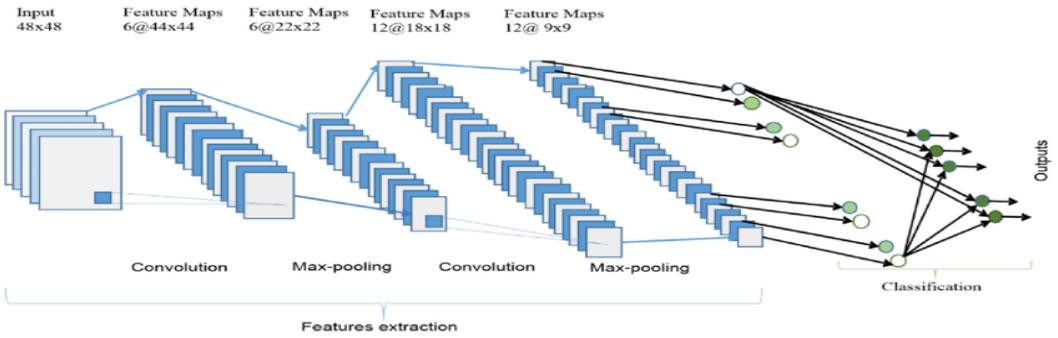


Figure 7 is showing layers of CNN

CNN is a special case of the neural network.

References below help us a lot to know more about CNN.

The CNN consists of one or more convolutional layers, often with a subsampling Layer, which are followed by one or more fully connected layers as in a standard Neural network.



The design of a CNN is motivated by the discovery of a visual mechanism, the visual cortex, in the brain.

The visual cortex contains a lot of cells that are responsible for detecting light in small, overlapping sub-regions of the visual field, which are called receptive fields. These cells act as local filters over the input space, and the more complex cells have larger receptive fields.

The convolution layer in CNN performs the function that is performed by the cells in the visual cortex. A typical CNN for recognizing traffic signs.

Each feature of a layer receives inputs from a set of features located in a small neighborhood in the previous layer called a local receptive field. With local receptive fields, features can extract elementary visual features, such as oriented edges, end-points, corners, etc., which are then combined by the higher layers.

In the traditional model of pattern/image recognition, a hand-designed feature extractor gathers relevant information from the input and eliminates irrelevant variabilities.

The extractor is followed by a trainable classifier, a standard neural network that classifies feature vectors into classes.

In a CNN, convolution layers play the role of feature extractor. But they are not hand-designed. Convolution filter kernel weights are decided on as part of the training process. Convolutional layers can extract the local features because they restrict the receptive fields of the hidden layers to be local.

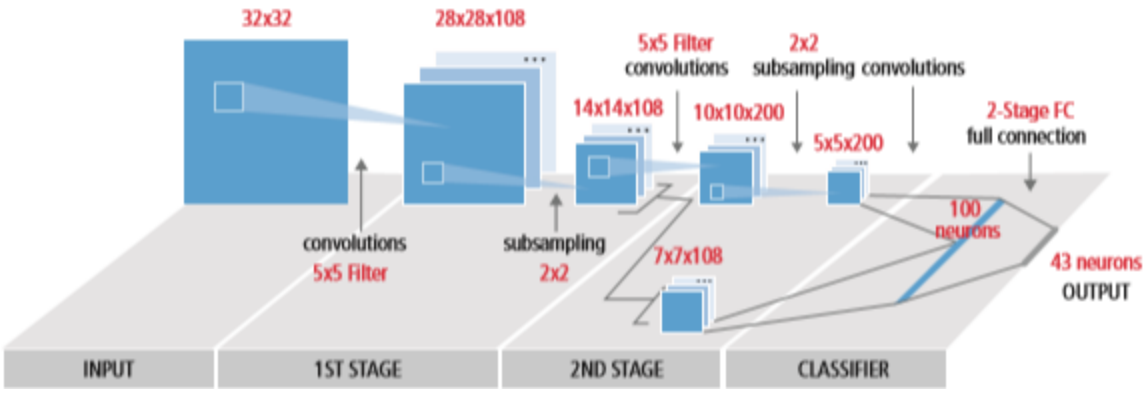


Figure 8 is showing a typical block diagram of CNN

CNN is used in a variety of areas, including image and pattern recognition, speech recognition, natural language processing, and video analysis.

There are several reasons that convolutional neural networks are becoming important. In traditional models for pattern recognition, feature extractors are hand-designed. In CNN, the weights of the convolutional layer being used for feature extraction as well as the fully connected layer being used for classification are determined during the training process.

The improved network structures of CNN lead to savings in memory requirements and computation complexity requirements and, at the same time, give better performance for applications where the input has local correlation (e.g., image and speech).

Large requirements of computational resources for training and evaluation of CNNs are sometimes met by graphic processing units (GPUs), DSPs, or other silicon architectures optimized for high throughput and low energy when executing the idiosyncratic patterns of CNN computation.

Advanced processors such as the ten silica Vision P5 DSP for Imaging and Computer Vision from Cadence have an almost ideal set of computation and memory resources required for running CNNs at high efficiency. In pattern and image recognition applications, the best possible correct detection rates (CDRs) have been achieved using CNN.

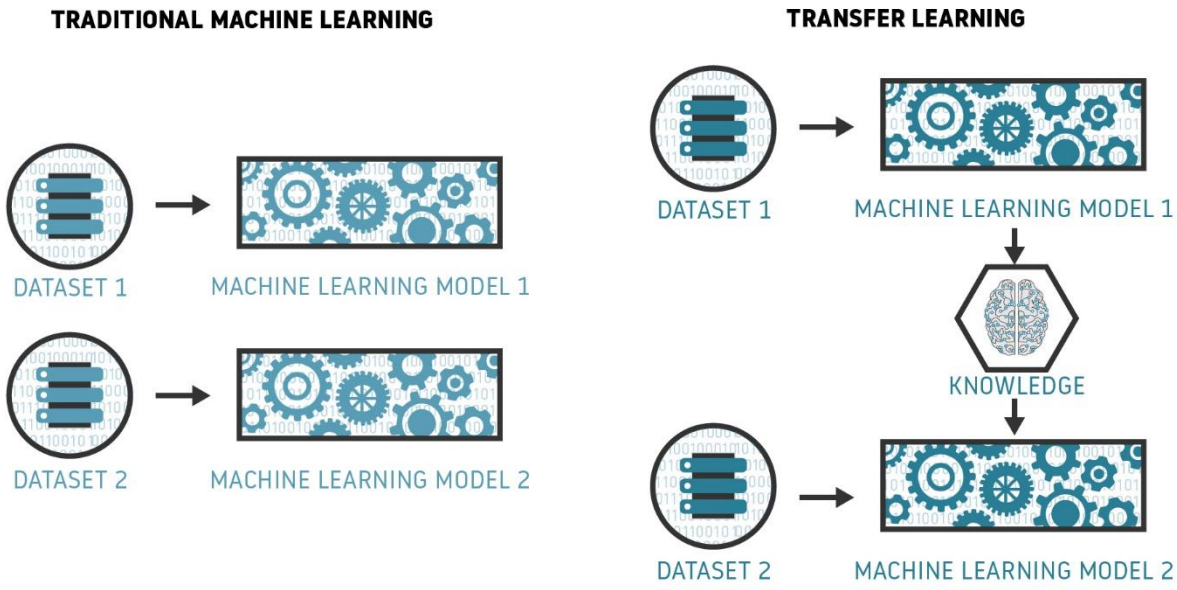
For example, CNN has achieved a CDR of 99.77% using the MNIST database of handwritten digits , a CDR of 97.47% with the NORB dataset of 3D objects, and a CDR of 97.6% on ~5600 images of more than 10 objects.

CNN not only gives the best performance compared to other detection algorithms, they even outperform humans in cases such as classifying objects into fine-grained categories such as the particular breed of dog or species of bird. typical vision algorithm pipeline, which consists of four stages: pre-processing the image, detecting regions of interest (ROI) that contain likely objects, object recognition, and vision decision making.

The pre-processing step is usually dependent on the details of the input, especially the camera system, and is often implemented in a hardwired unit outside the vision subsystem. The decision making at the end of the pipeline typically operates on recognized objects—it may make complex decisions, but it operates on much less data, so these decisions are not usually computationally hard or memory-intensive problems. The big challenge is in the object detection and recognition stages, where CNN is now having a wide impact and the following figure shows algorithm pipeline.

Understanding Transfer Learning

The first thing to remember here is that, transfer learning, is not a new concept that is very specific to deep learning. There is a stark difference between the traditional approach of building and training machine learning models and using a methodology following transfer learning principles.



Transfer Learning for Deep Learning

The strategies we discussed in the previous section are general approaches that can be applied towards machine learning techniques, which brings us to the question, can transfer learning be applied in the context of deep learning?

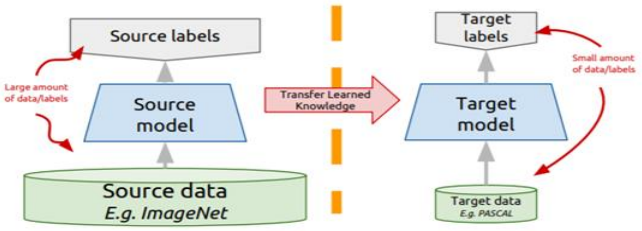
Transfer learning: idea

Instead of training a deep network from scratch for your task:

- Take a network trained on a different domain for a different **source task**
- Adapt it for your domain and your **target task**

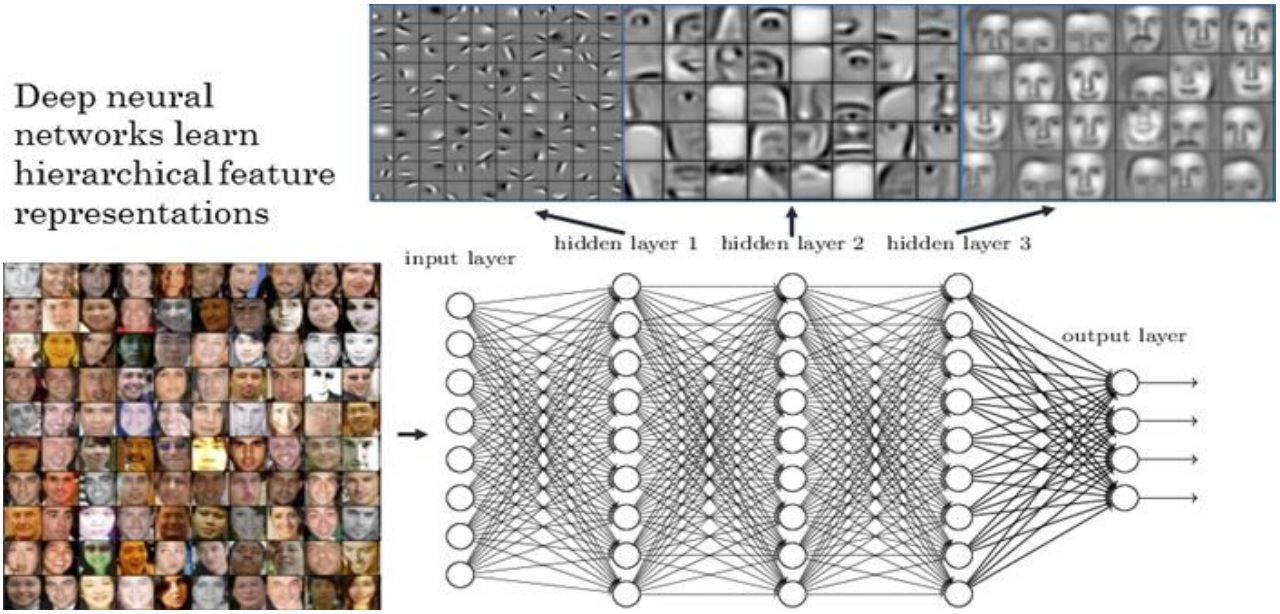
Variations:

- Same domain, different task
- Different domain, same task



Fine Tuning Off-the-shelf Pre-Trained Models

This is a more involved technique, where we do not just replace the final layer (for classification/regression), but we also selectively retrain some of the previous layers. Deep neural networks are highly configurable architectures with millions of hyperparameters. As discussed earlier, the initial layers have been seen to capture generic features, while the later ones focus more on the specific task at hand. An example is depicted in the following figure on a face-recognition problem, where initial lower layers of the network learn very generic features and the higher layers learn very task-specific features.



-Android application

Artificial intelligence (AI) techniques such as machine learning and deep learning. have grown rapidly in recent years in the context of computing with smart mobile phones that typically allows the devices to function in an intelligent manner.

Popular AI techniques include machine learning and deep learning methods, natural language processing, as well as knowledge representation and expert systems, can be used to make the target mobile applications intelligent and more effective.

Deep learning is a machine learning technique that teaches computers to do what comes naturally to humans: learn by example. Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost.

It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. It’s achieving results that were not possible before. One of the deep learning techinques is the CNN as previously mentioned.

In deep learning, a computer model learns to perform classification tasks directly from images, text, or sound. Deep learning models can achieve state-of-the-art accuracy, sometimes exceeding human-level performance. Models are trained by using a large set of labeled data and neural network architectures that contain many layers.

- why Android application ?

Smart phones phones act as a perfect device that can capture digital images or digital sound or write text on , using smart phone's camera can allow you to take a picture of a specific part of your skin that might contain a disease.

Then these images can be passed to CNN model that can accuritly predict the diseased part and provide informations of how it can be treated.

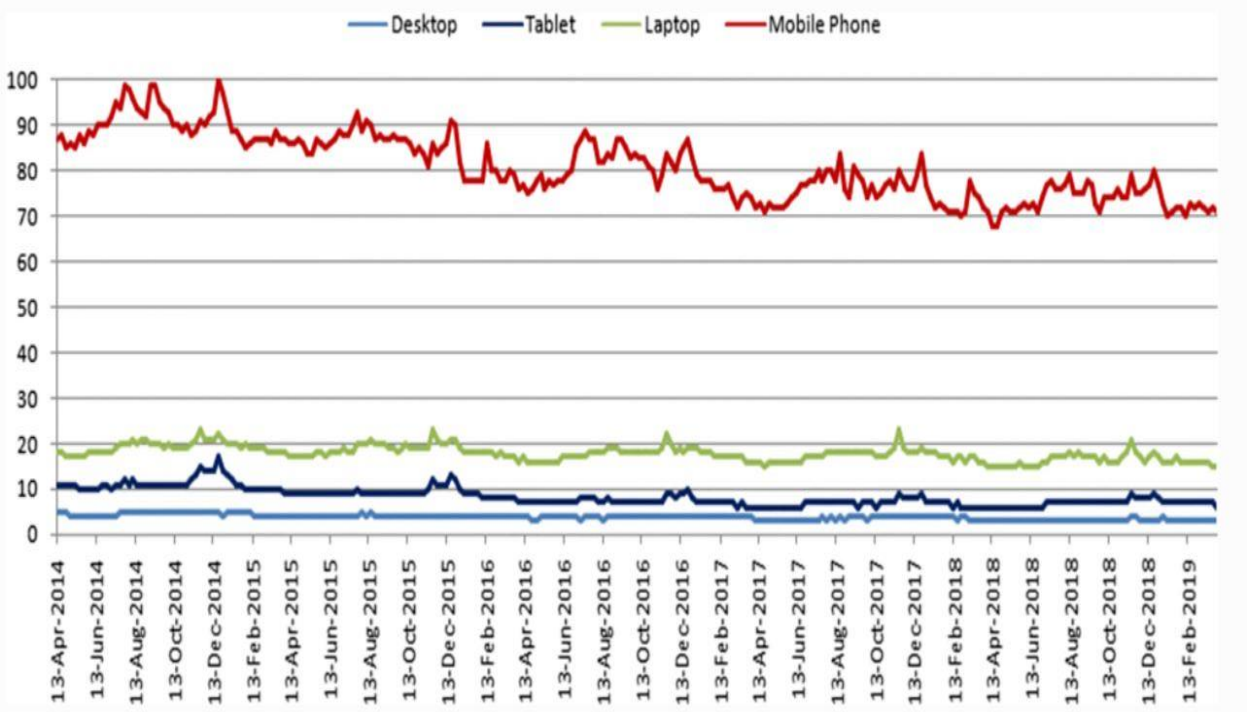
We can conclude from the previous passage that, we need to use images , text or sound to train our deep learning model wich is CNN in our project and not only to train the model but to use the model itself to give predictions of data mainly made from images or text or sound.

The traditional user of such digital images text and sound uses a personal computer but recently that's not the case anymore.

Due to the recent development of science and technology in the world, the smartphone industry has made exponential growth in the mobile phone application market . These devices are well known as one of the most important Internet-of-Things (IoT) devices as well, according to their diverse capabilities including data storage and processing .

Today's smartphone is also considered as "a next-generation, multi-functional cell phone that facilitates data processing as well as enhanced wireless connectivity", i.e., a combination of "a powerful cell phone" and a "wireless-enabled PDA" .

we have seen that users' interest on *"Mobile Phones"* is more and more than other platforms like *"Desktop Computer"*, *"Laptop Computer"* or *"Tablet Computer"* for the last five years from 2014 to 2019 according to Google Trends data .



This Figure shows *"Mobile Phones"* is more than other platforms like *Desktop Computer*, *Laptop* or *Tablet* for the last five years from 2014 to 2019 according to Google Trends data .

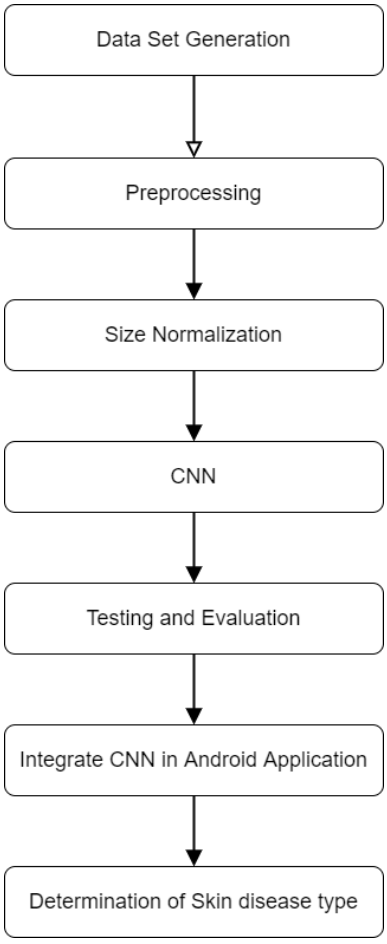
4- Implementation and Testing

4.1 Proposed System

4.1.1 Overview of the proposed system

The overall flowchart of our proposed skin disease detection system.

- Generating a large dataset of diseases' images from small one .
- In the pre-processing step, we use data augmentations techniques to generate larger dataset.
- Normalization divides the dataset into an equal proportion to avoid overfitting.
- CNN developing a CNN model to classify the image.
- Testing and evaluation CNN model.
- Integrate the CNN model into Android application to make it easier for the user to check his skin's health.



This Figure is showing system overview

4.1.2 Dataset generating

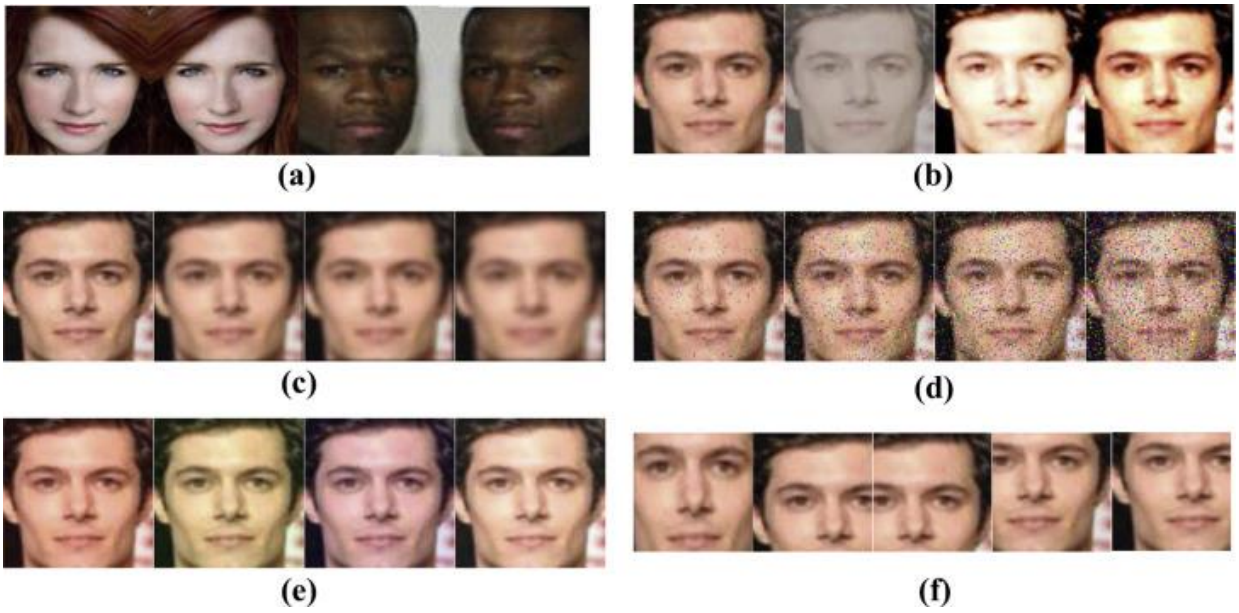
4.1.2.1 Data Augmentation

The prediction accuracy of the Supervised Deep Learning models is largely reliant on the amount and the diversity of data available during training. The relation between deep learning models and amount of training data required is analogous to that of the relation between rocket engines (deep learning models) and the huge amount of fuel (huge amounts of data) required for the rocket to complete its mission (success of the deep learning model).

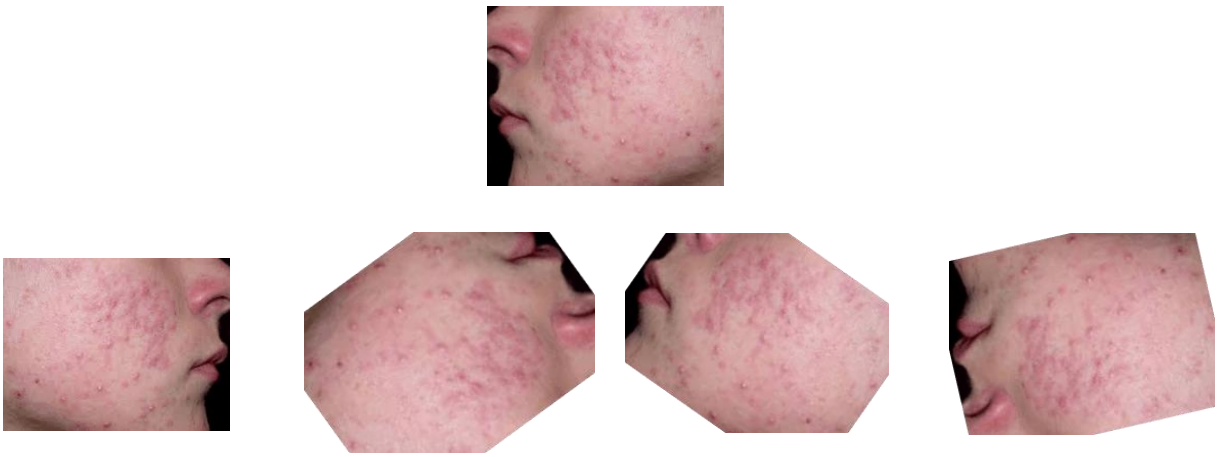
Deep learning models trained to achieve high performance on complex tasks generally have a large number of hidden neurons. As the number of hidden neurons increases, the number of trainable parameters also increases.

Oftentimes, when working on specific complex tasks such as classifying a weed from a crop, or identifying the novelty of a patient, it is very hard to get large amounts of data required to train the models.

applying different transformations on the available data to synthesize new data. This approach of synthesizing new data from the available data is referred to as ‘Data Augmentation’ as shown in the Figures below.



This Figure shows some Examples of data augmentations.

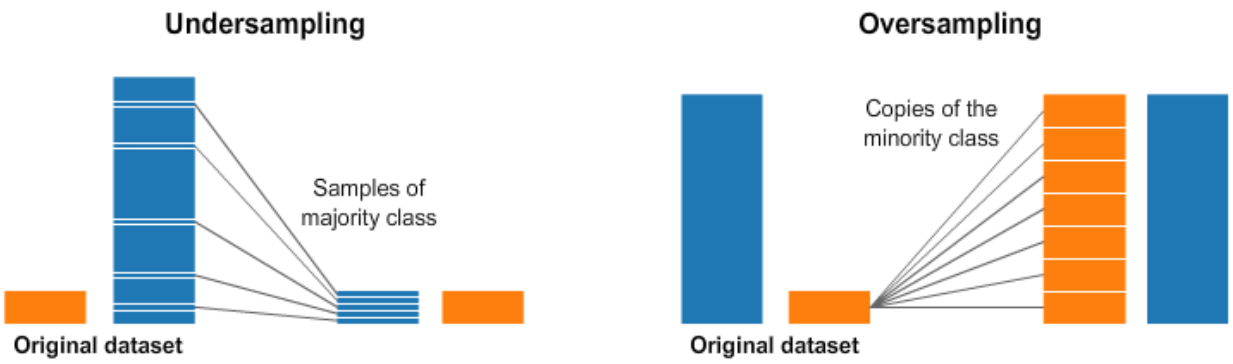


This Figure shows some examples of images augmentations

Data augmentation can be used to address both the requirements, the diversity of the training data, and the amount of data. Besides these two, augmented data can also be used to address the class imbalance problem in classification tasks.

Imbalanced data sets are a special case for classification problem where the class distribution is not uniform among the classes. Typically, they are composed by two classes: The majority (negative) class and the minority (positive) class.

These type of sets suppose a new challenging problem for Data Mining, since standard classification algorithms usually consider a balanced training set and this supposes a bias towards the majority class.



This Figure shows imbalanced data.

The questions that come to the mind are , Does data augmentation work? Do we really get better performance from the models when we use augmented data for training? Table below shows a few case studies indicating the effect of data augmentation on model performance for different applications.

Application	Performance without Augmentation	Performance with Augmentation	Augmentation method
Image classification	57%	78.6%	Simple Image based
Image classification	57%	85.7%	GAN based
NMT	11 BLEU	13.9 BLEU	Translation data augmentation
Text classification	79%	87%	Easy Data Augmentation

This Figure shows the performance after data augmentation.

4.1.3 Convolution Neural Network

4.1.3.1 Convolutional neural network

Convolutional Neural Networks are Artificial Neural Network models that are comparable to the visual cortex, which is a fully connected, stratified network with each layer providing regions of cells that are sentient of specific fields of vision. This type of network is specifically designed for making full use of data with good spatial relations.

The stratified topology of the convolutional neural network wheels out different properties of each layer, serialized as a collection of convolutional, activation, pooling, and fully connected layers, with the entire network comprising of at least one layer of convolution.

This convolutional layer takes advantage of the fact that input is made up of spatially related data, and have neurons arranged in 3 dimensions consisting of width, height, and depth of the activation volume.

Mathematically, this layer computes a dot product between the weights of local receptive fields in the input and the connected region in the input volume and produces another array of numbers known as activation map or feature map. Each local receptive field converges into a hidden neuron connected to it in the succeeding layer. Since the output of the convolution produces a linear transformation of the input, it does not satisfy the universal approximation theorem which insinuates that the representational power of the network is coerced with linearity.

Hence, for the network to comply with the universal approximator, the activation layer is required. The sole purpose of the activation layer is to infuse non-linearities in the network.

The output of this layer is then pooled or downsampled in the pooling layer to simplify the feature map produced by the preceding layers.

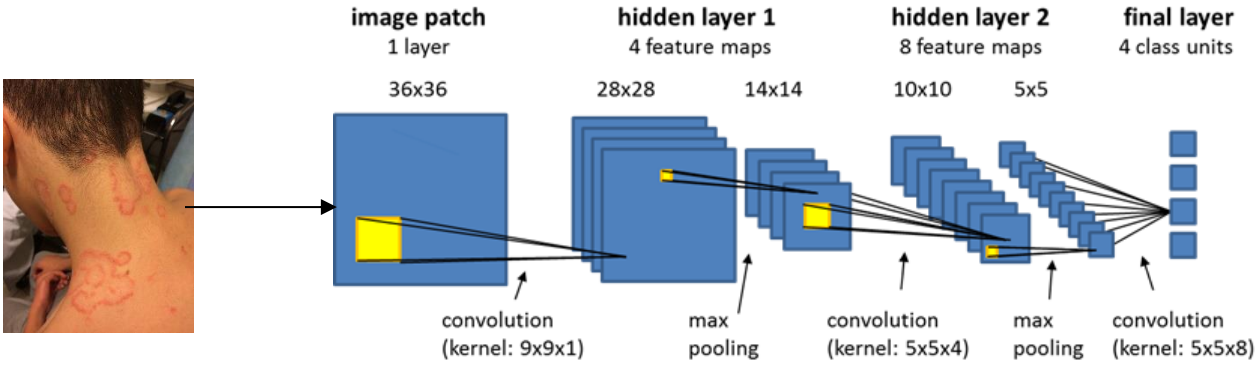
The pooling layer takes advantage of the fact that when a feature is extracted by the previous layers, the location of the feature in the feature map is not as important as its location compared to other detected features.

This reduces the spatial size of the representation, consequently reducing the computational cost of the entire network and abating overfitting as well.

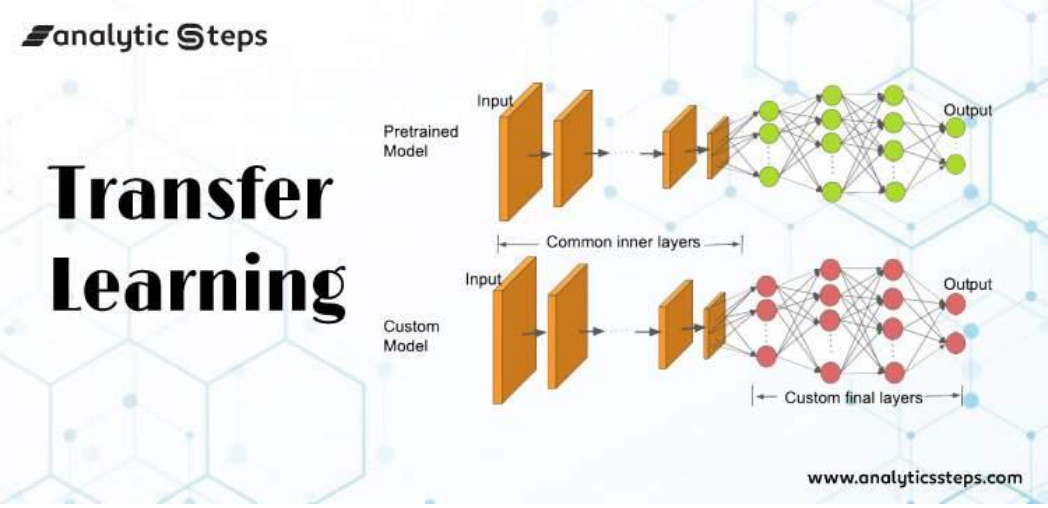
The entire composed stack is then consolidated into a fully connected layer, wherein each neuron is connected to every neuron in the preceding layer.

fully connected layer is usually one-dimensional and comprises all the labels that are to be classified. This layer outputs a score for each label of classification.

Since the convolutional networks are specifically designed to make use of spatial relations between the objects in the provided data, they perform better in regions of machine learning where the related spatial data needs to be manipulated, such as detection or recognition of objects. Also, as these convolutional networks explicitly assume that the inputs are images, it provides a streamlined function to encode the data and implement the network with immensely reduced parameters.

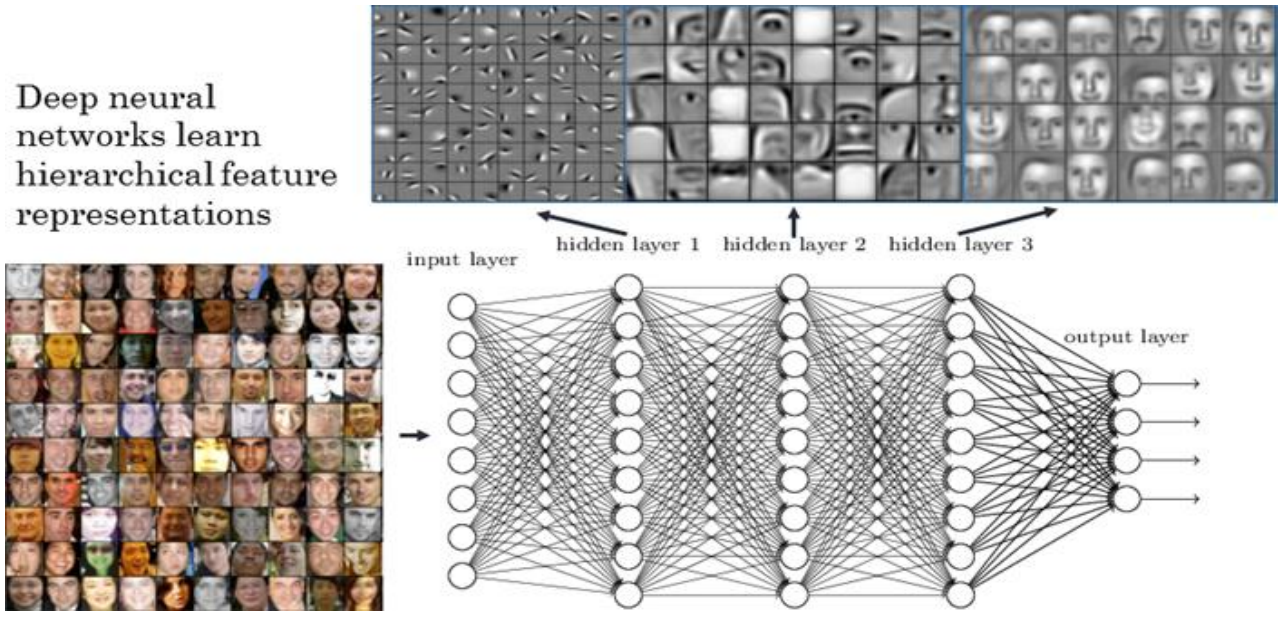


4.1.3.2 Transfer learning



Fine-tuning off-the-shelf Pre-Trained Models

This is a more involved technique, where we do not just replace the final layer (for classification/regression), but we also selectively retrain some of the previous layers. Deep neural networks are highly configurable architectures with various hyperparameters. As discussed earlier, the initial layers have been seen to capture generic features, while the later ones focus more on the specific task at hand. An example is depicted in the following figure on a face-recognition problem, where initial lower layers of the network learn very generic features and the higher layers learn very task-specific features.

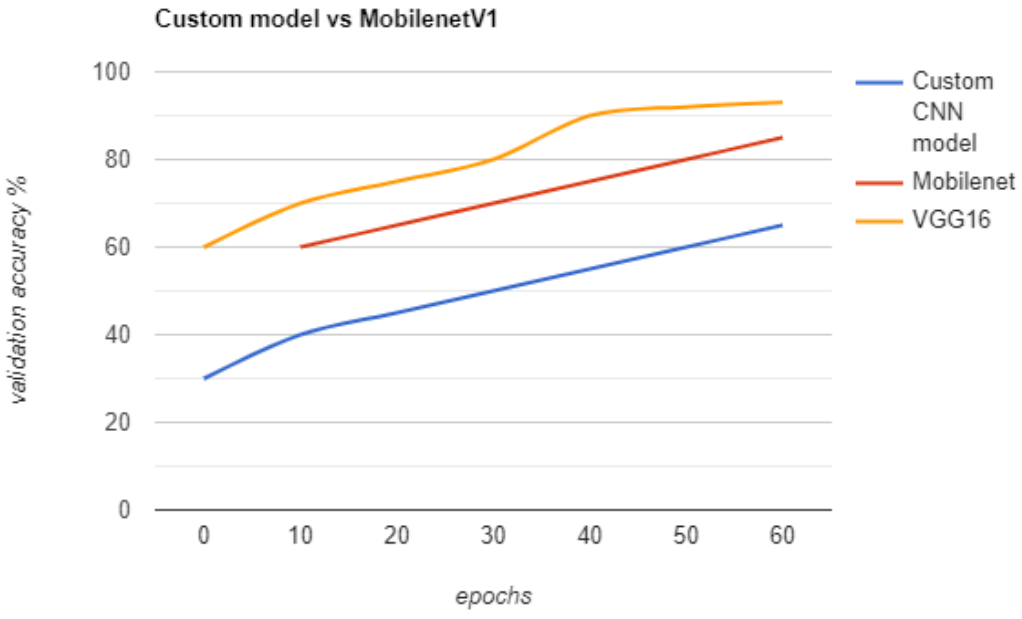


4.1.3.3 Select pre-train models

Selecting the most suitable Transfer Pretrained model to use, by using 3 models and train them with our data and measure its metrics. By training and validating , we found that VGG16 has higher validation accuracy among all the other pretrained models .

VGG16 reached to 93% but the size of the model was way too large to be used in An android application, vgg16 has about 130Million parameters. According to the previous information vgg16 architecture is very complex and large that had affected the size of the model .

so we had to look for another alternative ways to use the model inside android application. That's when tensorflowlite solution was used, but even converting the model to tensorflowlite extension the model was still large to be used inside android application so we had to use the second accurate pretrained model to implement.



This figure shows validation accuracy % of various pretrained models

4.1.3.4 MobileNet V1

What is mobile net :

The MobileNet model is based on depthwise separable convolutions which is a form of factorized convolutions which factorize a standard convolution into a depthwise convolution and a 1×1 convolution called a pointwise convolution. For MobileNets the depthwise convolution applies a single filter to each input channel. The pointwise convolution then applies a 1×1 convolution to combine the outputs the depthwise convolution. A standard convolution both filters and combines inputs into a new set of outputs in one step.

The depthwise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size. standard convolution is factorized into a depthwise convolution and a 1 × 1 pointwise convolution . A standard convolutional layer takes as input a DF × DF × M feature map F and produces a DF × DF × N feature map G where DF is the spatial width and height of a square input feature map1 , M is the number of input channels (input depth), DG is the spatial width and height of a square output feature map and N is the number of output channel (output depth). The standard convolutional layer is parameterized by convolution kernel K of size DK ×DK ×M×N where DK is the spatial dimension of the kernel assumed to be square and M is number of input channels and N is the number of output channels as defined previously. The output feature map for standard convolution assuming stride one and padding is computed as:

$$G_{k,l,n} = X_{i,j,m} K_{i,j,m,n} \cdot F_{k+i-1,l+j-1,m} \quad (1)$$

Standard convolutions have the computational cost of:

$$D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F \quad (2)$$

where the computational cost depends multiplicatively on the number of input channels M, the number of output channels N the kernel size Dk × Dk and the feature map size DF × DF . MobileNet models address each of these terms and their interactions.

First it uses depthwise separable convolutions to break the interaction between the number of output channels and the size of the kernel.

The standard convolution operation has the effect of filtering features based on the convolutional kernels and combining features in order to produce a new representation.

The filtering and combination steps can be split into two steps via the use of factorized convolutions called depthwise separable convolutions for substantial reduction in computational cost.

Depthwise separable convolution are made up of two layers: depthwise convolutions and pointwise convolutions.

We use depthwise convolutions to apply a single filter per each input channel (input depth). Pointwise convolution, a simple 1×1 convolution, is then used to create a linear combination of the output of the depthwise layer.

MobileNets use both batchnorm and ReLU nonlinearities for both layers. Depthwise convolution with one filter per input channel (input depth) can be written as:

$$\hat{G}_{k,l,m} = X_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m} \quad (3)$$

where $\hat{K}$ is the depthwise convolutional kernel of size $D_K \times D_K \times M$ where the mth filter in $\hat{K}$ is applied to the mth channel in F to produce the mth channel of the filtered output feature map $\hat{G}$. Depthwise convolution has a computational cost of:

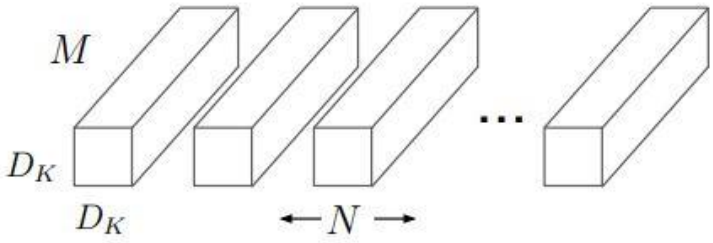
$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F \quad (4)$$

Depthwise convolution is extremely efficient relative to standard convolution. However it only filters input channels, it does not combine them to create new features. So an additional layer that computes a linear combination of the output of depthwise convolution via 1 × 1 convolution is needed in order to generate these new features. The combination of depthwise convolution and 1 × 1 (pointwise) convolution is called depthwise separable convolution which was originally introduced previously. Depthwise separable convolutions cost:

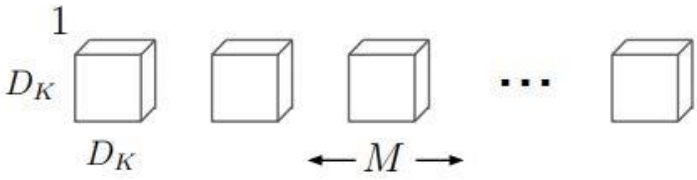
DK · DK · M · DF · DF + M · N · DF · DF (5) which is the sum of the depthwise and 1 × 1 pointwise convolutions. By expressing convolution as a two step process of filtering and combining we get a reduction in computation of:

DK · DK · M · DF · DF + M · N · DF · DF DK · DK · M · N · DF · DF = 1 N + 1 D2 K

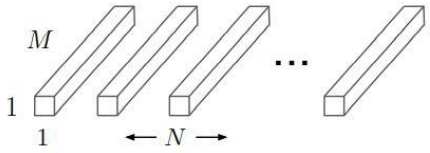
MobileNet uses 3 × 3 depthwise separable convolutions which uses between 8 to 9 times less computation than standard convolutions at only a small reduction in accuracy as seen in Section 4. Additional factorization in spatial dimension such as in [16, 31] does not save much additional computation as very little computation is spent in depthwise convolutions.



(a) Standard Convolution Filters



(b) Depthwise Convolutional Filters



(c) 1 × 1 Convolutional Filters called Pointwise Convolution in the context of Depthwise Separable Convolution

The MobileNet structure is built on depthwise separable convolutions as mentioned in the previous section except for the first layer which is a full convolution. By defining the network in such simple terms we are able to easily explore network topologies to find a good network. The MobileNet architecture is defined in Table 1. All layers are followed by a batchnorm [13] and ReLU nonlinearity with the exception of the final fully connected layer which has no nonlinearity and feeds into a softmax layer for classification. Figure 3 contrasts a layer with regular convolutions, batchnorm and ReLU nonlinearity to the factorized layer with depthwise convolution, 1 × 1 pointwise convolution as well as batchnorm and ReLU after each convolutional layer. Down sampling is handled with strided convolution in the depthwise convolutions as well as in the first layer. A final average pooling reduces the spatial resolution to 1 before the fully connected layer. Counting depthwise and pointwise convolutions as separate layers, It is also important to make sure these operations can be efficiently implementable.

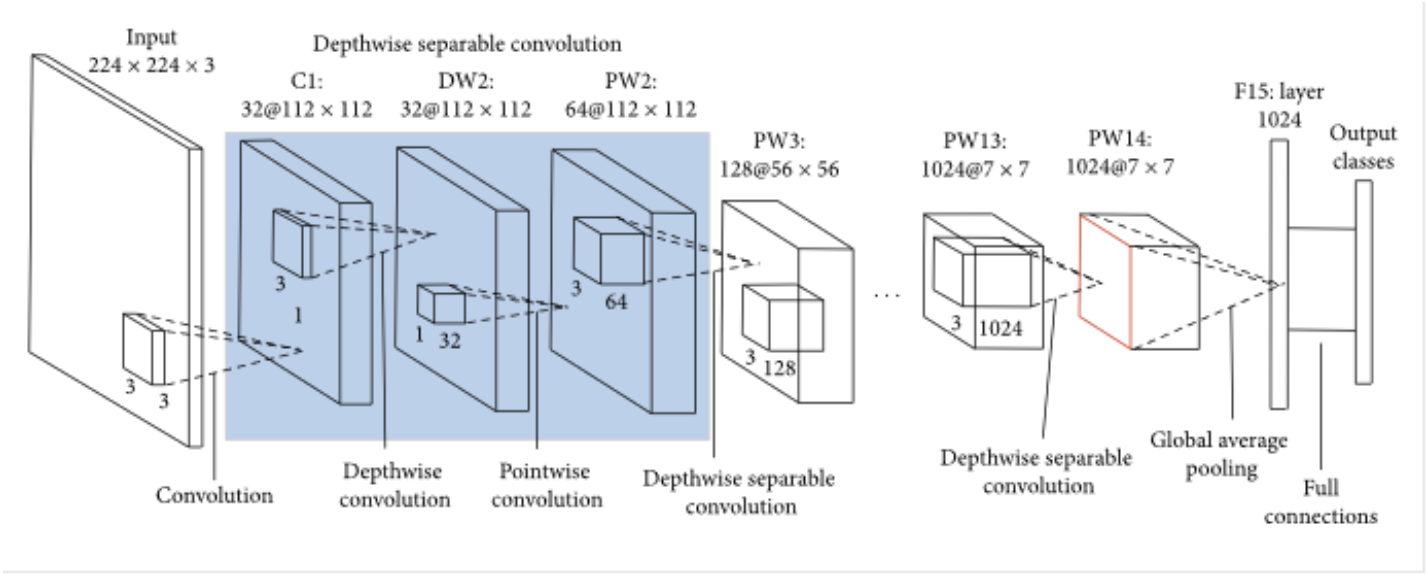


Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

4.1.4 Test and Evaluation

During the training phase, we divide dataset to 70% training and 30% testing and calculate validation accuracy in each epoch. The validation accuracy of our model reached 87% during the training phase.

4.1.5 Integrate CNN model into Android application

Integration of the pretrained convolutional neural network model have been done using tensorflow lite Framework , tensorflow lite is an open source deep learning Framework used to send data from an android application using mobile phone's resources such as camera , sound .etc to the trained CNN model for image classification. Due to the using of tensorflow lite, we were able to integrate the model with Native android application.

4.1.6 Classification of skin Disease type

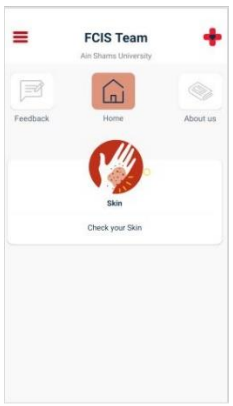
After feeding an image into our model we calculate from the output layer from our model which neuron has the maximum percentage that might be the right disease type and return the identified disease class as text .

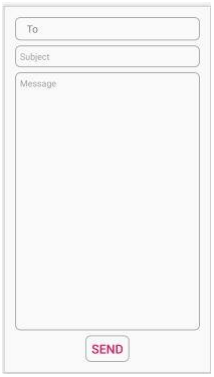
4.2 UI design

By Using Android studio , java , we have developed native android mobile application .

Implement

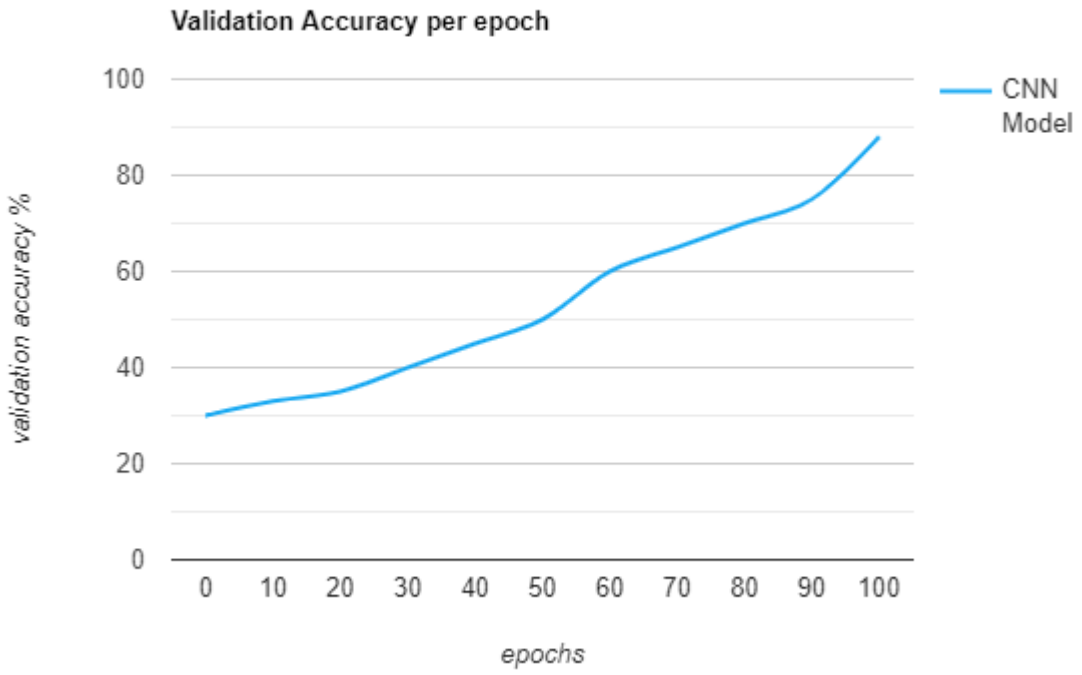
1. Splash
2. Main activity
3. Skin diagnose activity
4. About us
5. Feedback
6. WebView activity
7. Camera activity
8. Image picker activity





4.3 Testing procedures

During the training phase, we divide dataset to 70% training and 30% testing and calculate validation accuracy in each epoch, Then create a confusion matrix to indicate the performance of our model by using patient images. Test as a real-life application we developed an android application to that use our model and give accurate results to classify the disease .



5- User Manual

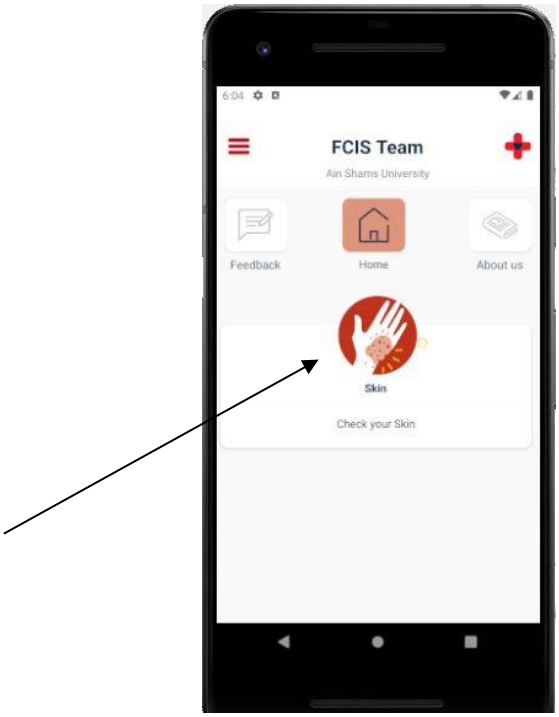
5.1 Tools should be available

1 Python programming language
2 Keras deep learning framework
3 Anaconda distribution platform
4 Spyder integrated development environment
5 Android studio
6 Java programming language
7 Tensorflow lite framewrok

5.2 Desktop application User Manual

we develop an Android application by using android studio to integrate our model in. Using the mobile's camera to capture the diseased skin part image, then send it to our model to do the classification process and then receive classification results as text then the user can view details about the disease and how to cure it.

1. Home menu



The home menu components:

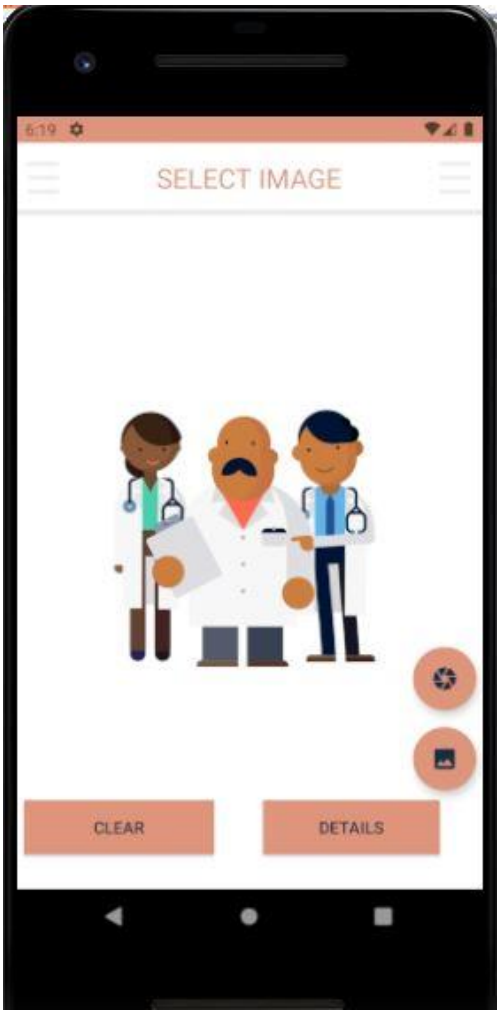
1- Feedback
2- About us
3- Check skin button

This is the home menu which is the first menu that the user could interact with

The user can choose various functionalities to do from this menu :

- The user click on check your skin button that will take the user to the screen where he can check his skin condition by taking a picture from his camera or use a photo from his gallery .
- The user can click on feedback to go to the feedback screen so he can rate our application .
- Home button is indicating the current opend menu.

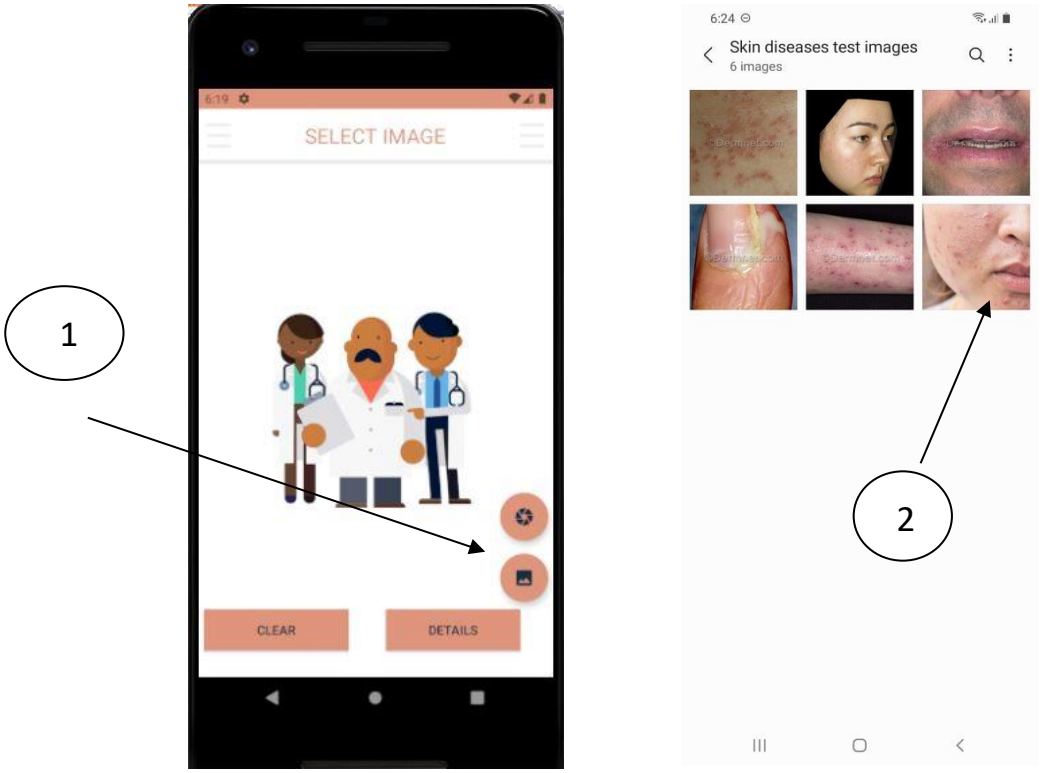
2. Check skin screen

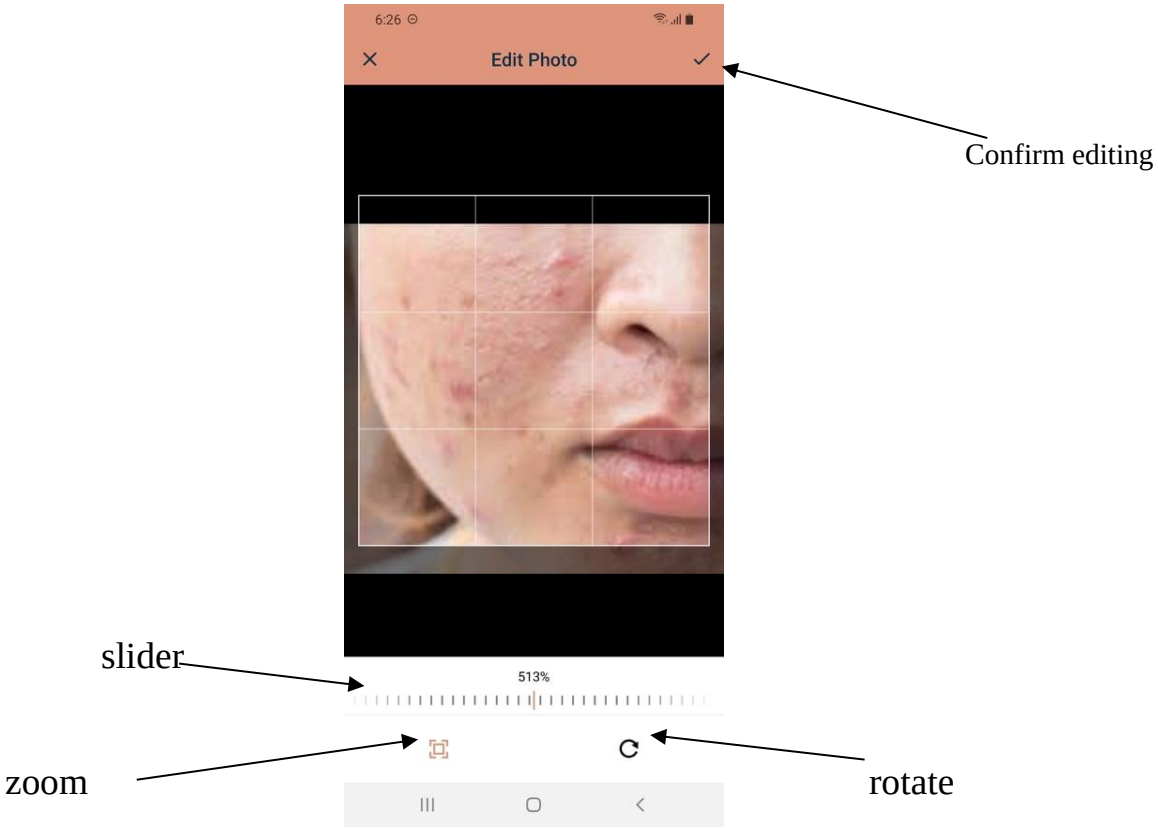


Check skin screen components:

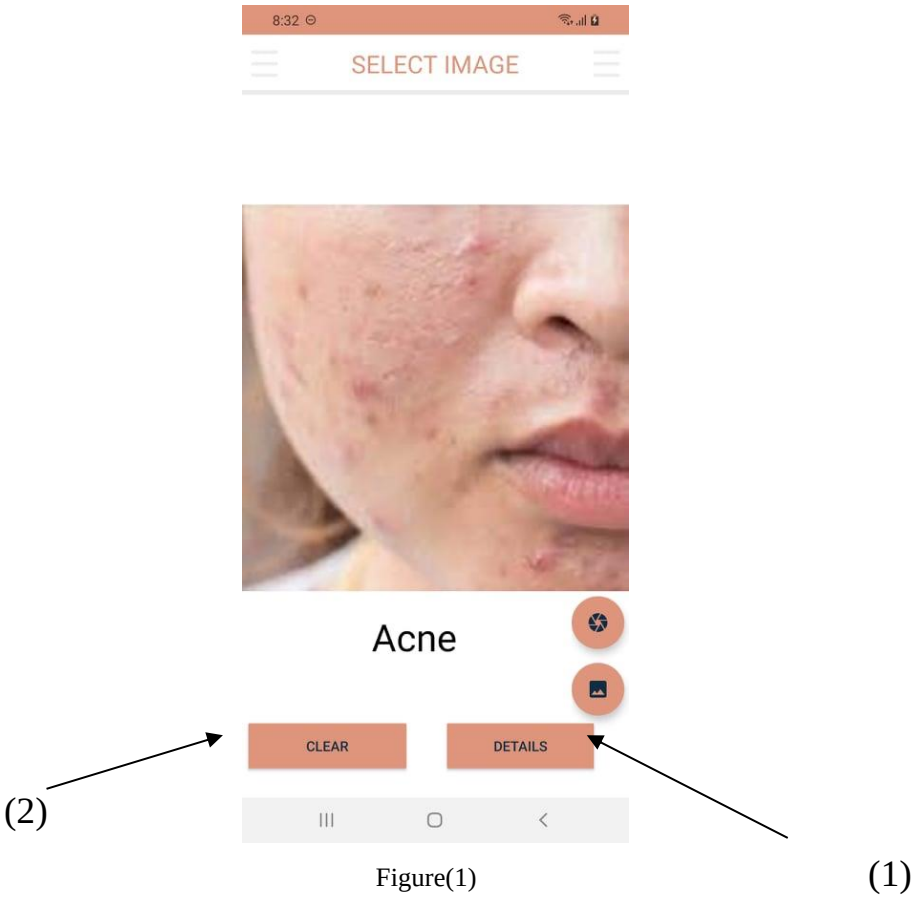
1- Details button
2- Clear
3- Camera button
4- Gallery button

The user clicks on gallery button to pick an image from his gallery to be diagnosed by our application:





- After the user picks the desired image to be diagnosed he can do some basic transformations and editing to the image by rotating the images , zooming in and out.
- The use can do so by clicking on what he wants to do zoom or rotate then he can use the slider to choose the desired scale.

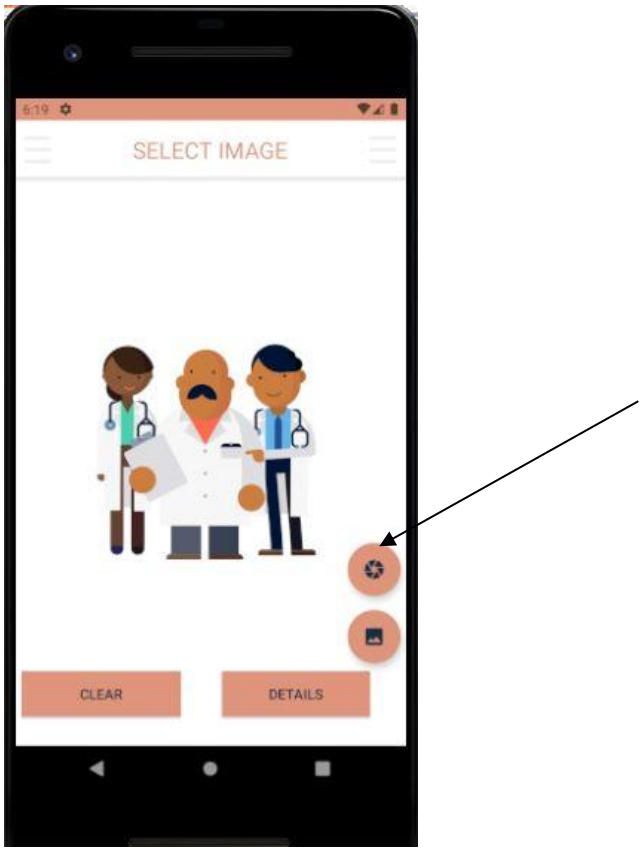


Figure(1)

- The name of the diagnosed disease is then given back to the user as a text . the user can click on details to get all the information about the disease (1) and how to cure it and prevent it from authenticated webpages in figure (2).

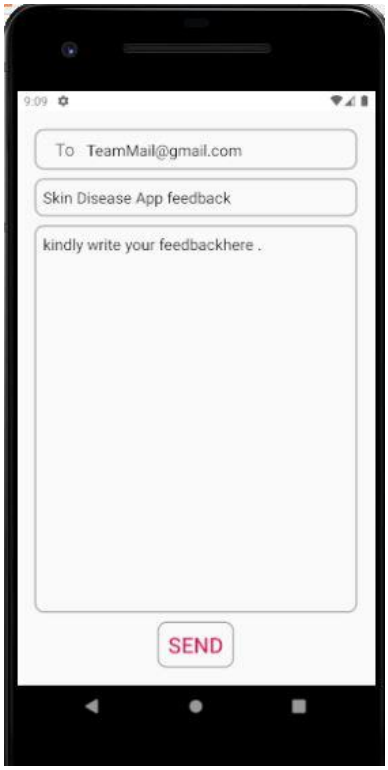
Figure (2)

- The user can clear the picture and repeat the previous process to select another image as shown int the previous page in figure(1).



- After the screen is cleared the user can use another images at his preference or use the camera to capture an image to get diagnosed.
- After capturing the image the user can still do the same edits to the image as zooming or rotating that was previously shown.

3. Feedback screen



- Going back to the home screen the user can click on feedback button and send us a feedback using an e-mail .

4. About us

- This page contain all the details and informations about our team and supervisiosors.

6- Conclusion and Future Work

6.1 Conclusion

Our proposed system is based on a modern solution type of deep learning CNN that insure Performing the process as fast and as robust as possible. The basic techniques utilized in our proposed system include image dataset Augmentation, Convolution operations, features extraction, and finally classify the image based on skin diseases dataset. The experimental results demonstrate that the proposed method can Recognize skin diseases with high accuracy reaches 87% and in a short time.

6.2 Future Work

In future work, we will address the issue of large pretrained models that require large storage that can exceed 0.5GB , this type of models can't be used without creating an API and deploy our model on it and then it can be used from the cloud and not affecting the mobile storage at all .

Another features will be added in the future work such as adding more diseases to our model to be identified , increase the accuracy of the current model so that it can pass 95%.

More features will be added in the android application such as map services where the patients can locate nearby pharmacies or hospitals which can provide health care to them.

Creating profile's for patients and their medical records, this can help us in the future to identify specific diseases and help us to provide the best health services to the patient.

References

1. Arifin, S., Kibria, G., Firoze, A., Amini, A., & Yan, H. (2012) “Dermatological Disease Diagnosis Using Color-Skin Images.” Xian: International Conference on Machine Learning and Cybernetics.
2. Yasir, R., Rahman, A., & Ahmed, N. (2014) “Dermatological Disease Detection using Image Processing and Artificial Neural Network. “Dhaka: International Conference on Electrical and Computer Engineering.
3. Santy, A., & Joseph, R. (2015) “Segmentation Methods for Computer Aided Melanoma Detection.” Global Conference on Communication Technologies.
4. Zeljkovic, V., Druzgalski, C., Bojic-Minic, S., Tameze, C., & Mayorga, P. (2015) “ Supplemental Melanoma Diagnosis for Darker Skin Complexion Gradients.” Pan American Health Care Exchanges
5. Suganya R. (2016) “An Automated Computer Aided Diagnosis of Skin Lesions Detection and Classification for Dermoscopy Images.” International Conference on Recent Trends in Information Technology.
6. Alam, N., Munia, T., Tavakolian, K., Vasefi, V., MacKinnon, N., & Fazel-Rezai, R. (2016) “Automatic Detection and Severity Measurement of Eczema Using Image Processing.” IEEE.
7. Kumar, V., Kumar, S., & Saboo, V. (2016) “Dermatological Disease Detection Using Image Processing and Machine Learning.” IEEE.
8. Krizhevsky, A., ILYA, S., & Geoffrey, E. (2012) “ImageNet Classification with Deep Convolutional Neural Networks.” Advances in Neural Information Processing Systems.
9. Cristianini, N., Shawe, J., “Support Vector Machines”, 2000.
10. SOMMERVILLE, I., “Software Engineering”. 9th .2011.
11. M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow. org, 1, 2015
12. M. Wang, B. Liu, and H. Foroosh. Factorized convolutional neural networks. arXiv preprint arXiv:1608.04337, 2016.