

DELHI TECHNOLOGICAL UNIVERSITY



DATABASE MANAGEMENT SYSTEM

CLASS PROJECT REPORT

Submitted To:

MANOJ SETHI

Submitted by:-

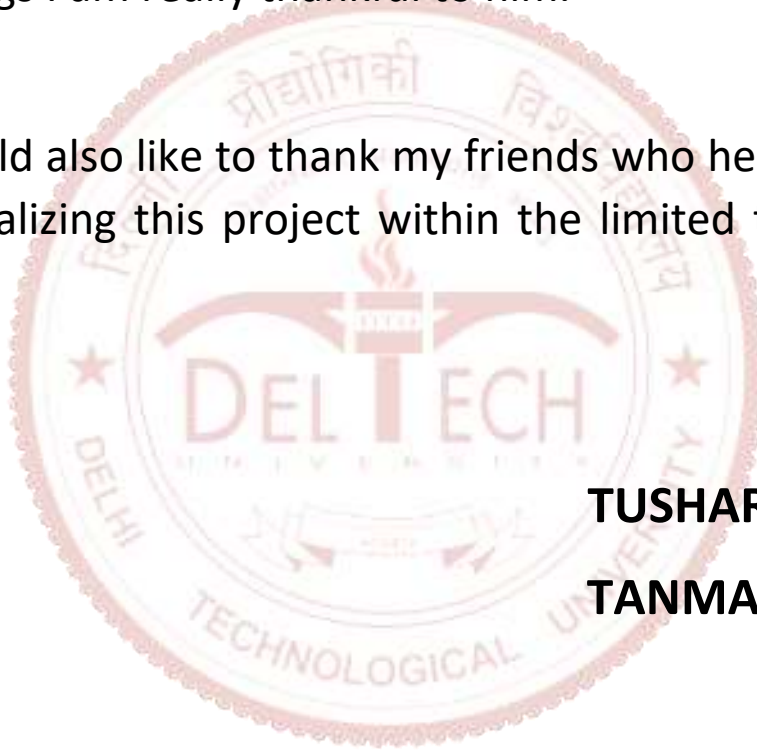
Tushar Kumar 2K19/CO/413

Tanmay Gupta 2K19/CO/406

ACKNOWLEDGEMENT

We would like to express my special thanks of gratitude to my teacher Mr Manoj Sethi who gave me the golden opportunity to do this innovative project on a very interesting topic “***Skin Disease Classification using Deep Learning***”, which also helped me in doing a lot of Research and I came to know about so many new things I am really thankful to him.

Secondly I would also like to thank my friends who helped me a lot in finalizing this project within the limited time frame.



TUSHAR KUMAR

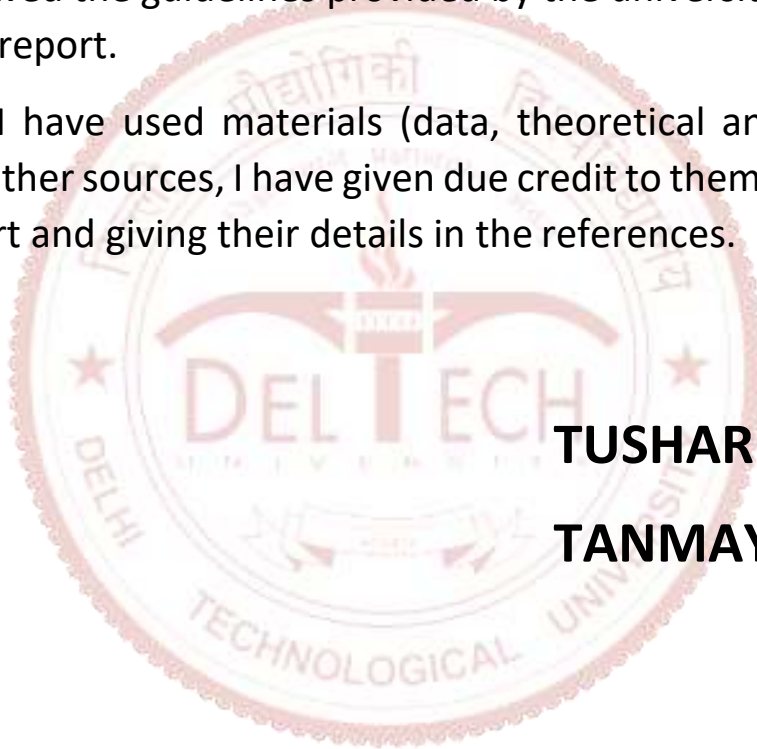
TANMAY GUPTA

DECLARATION

I undersigned solemnly declare that the project report is based on my own work carried out during the course of our study under the supervision of Mr Manoj Sethi.

I further certify that

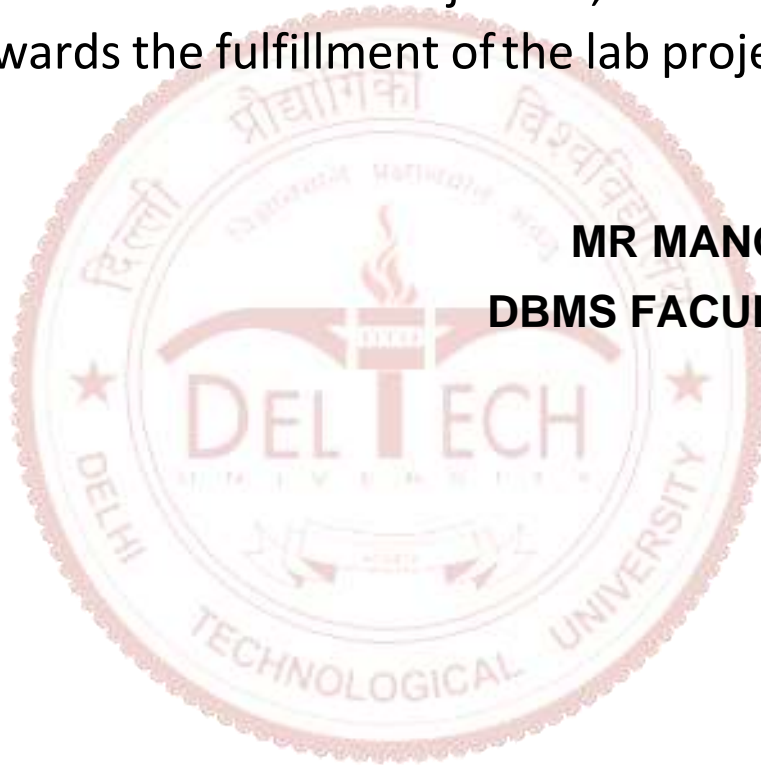
- I. The work contained in the report is original.
- II. I have followed the guidelines provided by the university in writing the report.
- III. Whenever I have used materials (data, theoretical analysis, and text) from other sources, I have given due credit to them in the text of the report and giving their details in the references.



TUSHAR KUMAR
TANMAY GUPTA

CERTIFICATE

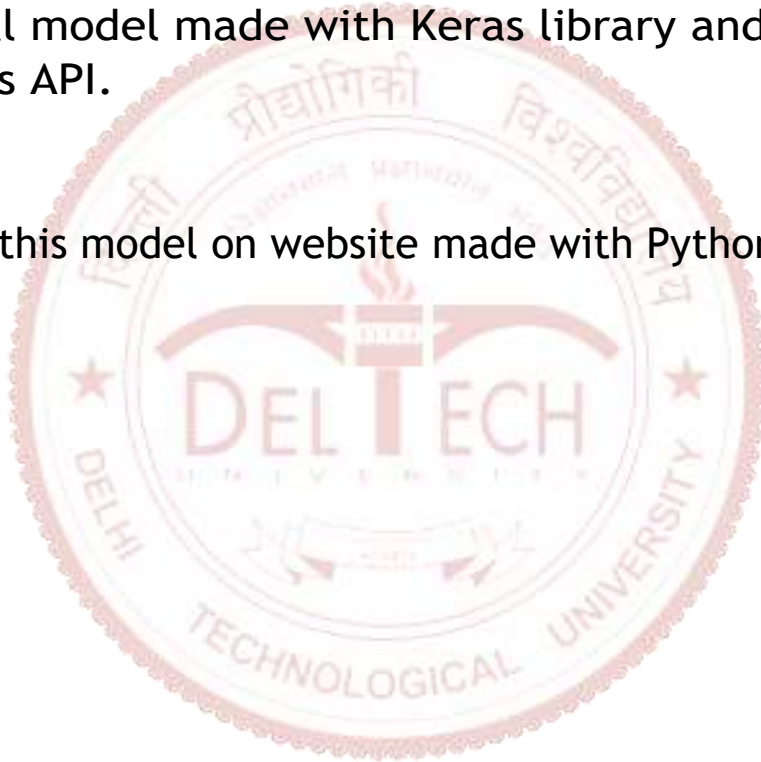
This is to certify that **TUSHAR KUMAR 2K19/CO/413 AND TANMAY GUPTA 2K19/CO/406** B.Tech Computer Engineering student of **DELHI TECHNOLOGICAL UNIVERSITY, DELHI** has done our work on this project, under the guidance of Mr Manoj Sethi, DBMS Faculty, DTU, Delhi towards the fulfillment of the lab project work.



MR MANOJ SETHI
DBMS FACULTY, DTU

ABSTRACT

- The project focuses on predicting the allergic reaction, understanding the pattern during the training in the dataset, Model architecture is specifically designed to predict to disease with very high accuracy.
- Machine Learning Model is made in python, it is a sequential model made with Keras library and saved with keras API.
- Deploying this model on website made with Python-Flask



CONTENTS

1. ABOUT

1.1 DESCRIPTION

2. DATA

2.1 DATA ANALYSIS

2.2 DATA CLEANSING

3. TRAINING AND ANALYSIS

3.1 THEORY

3.2 HYPER-PARAMETER

3.3 MODEL TRAINING

3.4 COMPARISON

3.5 ANALYSIS

4. DEPLOYMENT

4.1 DEPLOYING ON A WEBSITE

4.2 BACKEND USING FLASK

4.3 ADDITIONAL DATABASE

5. CONCLUSION AND REFERENCES

5.1 CONCLUSION

5.2 REFERENCES



INTRODUCTION

Description

The basic idea is to implementing a Deep network which can predict the type of skin infection.

This model can predict 18 different types of skin diseases.

- Acne and Rosacea
- Actinic Keratosis Basal Cell Carcinoma and other Malignant Lesions
- Atopic Dermatitis
- Bullous Disease
- Eczema
- Exanthems and Drug Eruptions
- Hair Loss Photos Alopecia and other Hair Diseases
- Herpes HPV and other STDs
- Light Diseases and Disorders of Pigmentation
- Lupus and other Connective Tissue diseases
- Melanoma Skin Cancer Nevi and Moles
- Nail Fungus and other Nail Disease
- Psoriasis pictures Lichen Planus and related diseases
- Seborrheic Keratoses and other Benign Tumors
- Tinea Ringworm Candidiasis and other Fungal Infections
- Urticaria Hives
- Vascular Tumors
- Warts Molluscum and other Viral Infections

DATA ANALYSIS AND CLEANSING

This is a huge dataset it contains almost 17064 images (after cleansing)

We got this dataset from one of the Kaggle's challenge, where the highest accuracy achieved with this dataset was around 20%.

Our main focus is mainly on increasing the accuracy, we have tried all the technique to clean our data and after that also did some hand-engineering to make our architecture better to get accurate output.

- There were 23 diseases in the dataset before.
- Now there are 18 diseases because samples picture of those 5 diseases were such waste like some body scan, x-rays, etc.
- Cleaned other samples pictures.
- We have selected some pictures from each folder which were appropriate for training to get better result.
- Performed same steps for Test dataset.

This dataset is very large and vulgar too, as we belong to a technical lane, so it is quite harder for us to remove all the unwanted pictures manually. That's why the accuracy of this model is around 65-70% only.

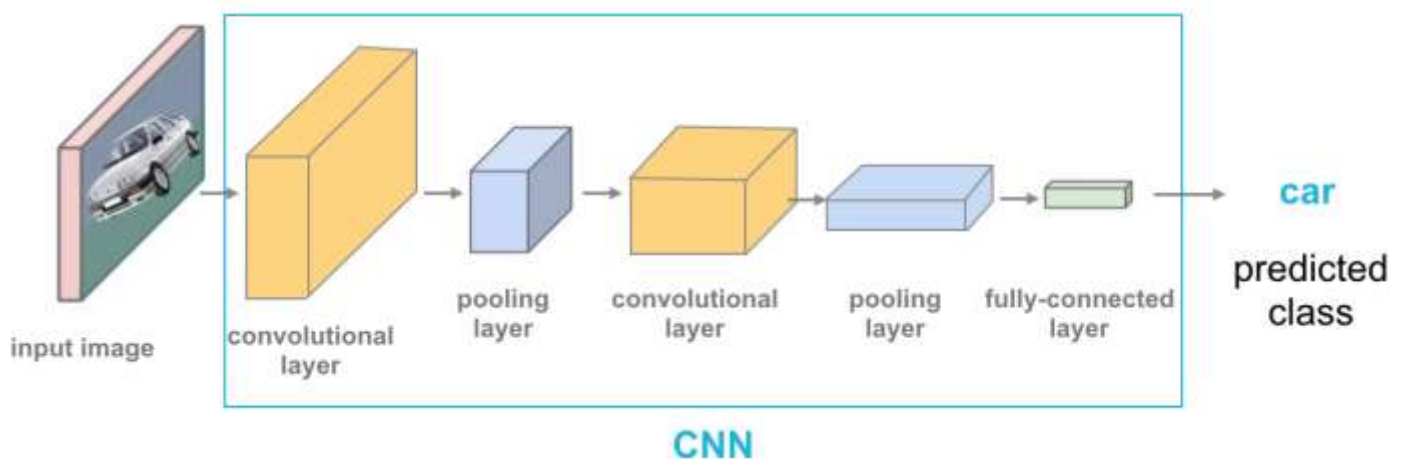
TRAINING AND ANALYSIS

Final model is conv net based architecture model. Deep learning has many techniques to mimic the working of human brain that is artificial intelligence.

We have trained this dataset on **CNN, Resnet50 and Resnet101** deep network.

Convolutional Neural Network :- CNN is an advanced and high-potential type of the classic artificial neural network model. It is built for tackling higher complexity, preprocessing, and data compilation. It takes reference from the order of arrangement of neurons present in the visual cortex of an animal brain.

- Convolutional layer, pooling layer, Activation and Batch Normalization.
- Fully connected Neural Network which produces output.

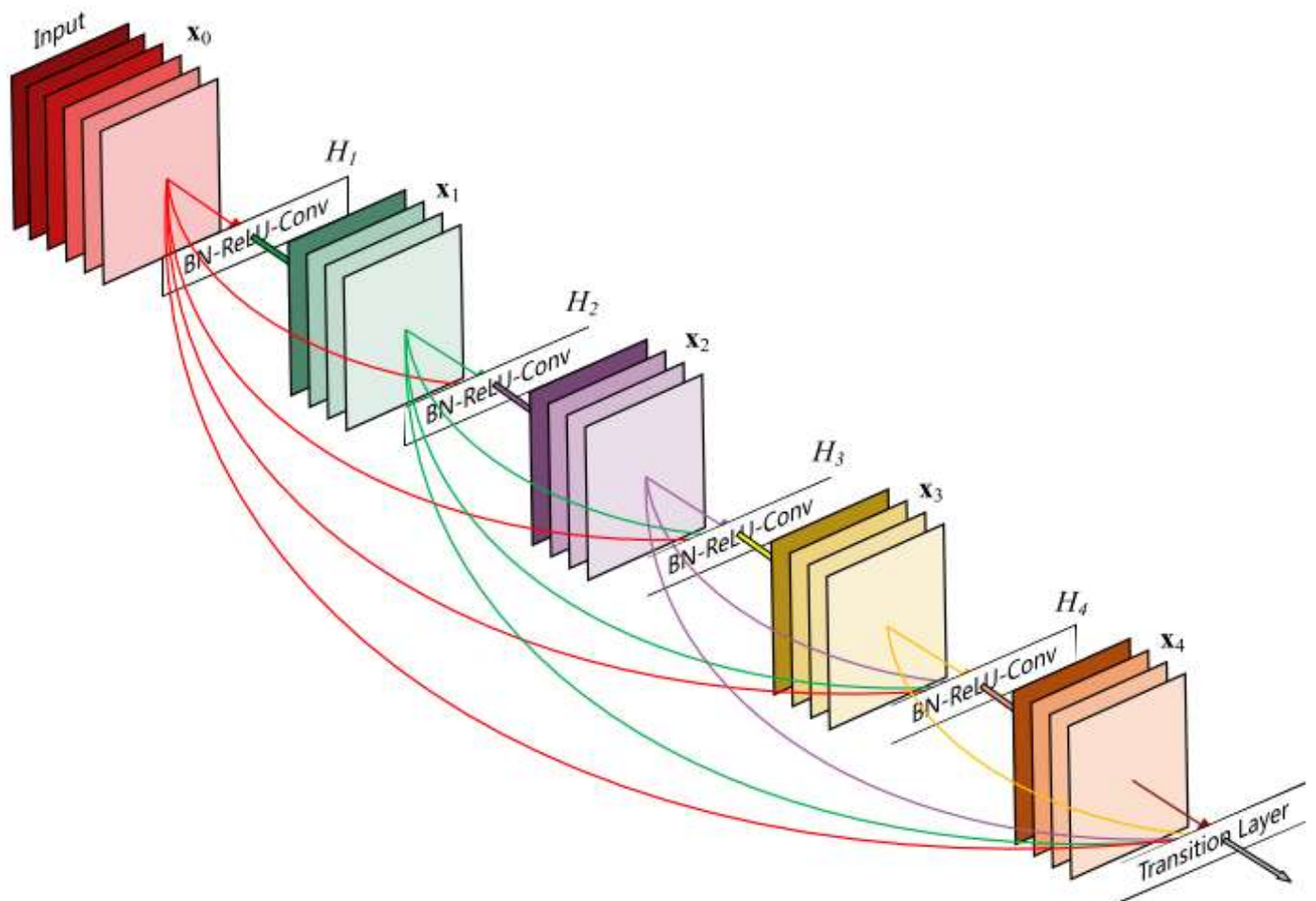


Residual Neural Network :- Net ResNet, short for Residual Network is a specific type of neural network. “**Deep Residual Learning for Image Recognition**” is based on skip connections, basic idea of skip connections is when a network is too deep then layers by layers the picture lost its initial features so we add the initial input to the input after few layers as shown in figure.

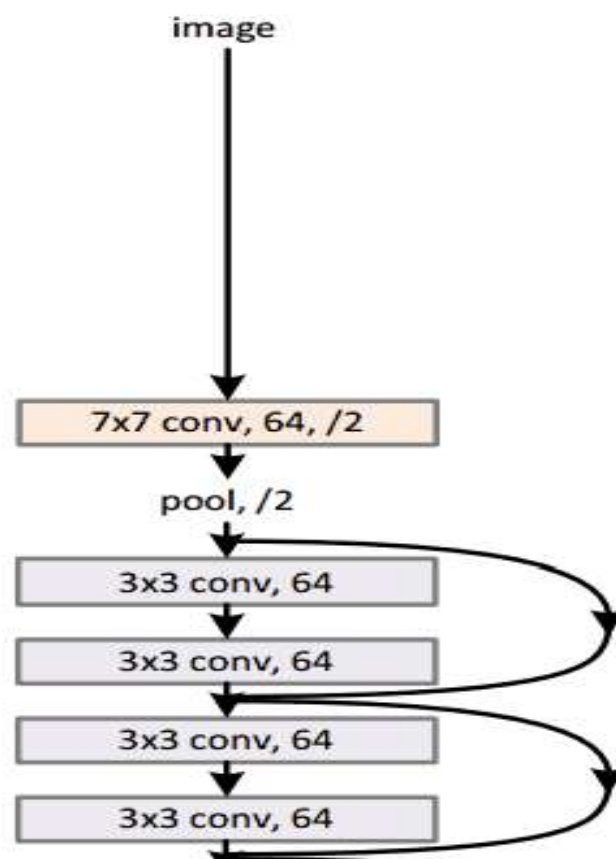
Hyper-Parameters

Hyper-parameter are the parameters that are independent of Data.

- Stride, Pool, Filters, etc.
- Learning rate, gamma, beta etc.



34-layer residual



1. In the very beginning, we have trained our model on simple CNN network which achieved the accuracy around 35%.

```
model=Sequential()

model.add(Conv2D(96,(11,11),strides=4,input_shape=x.shape[1:]))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(3,3),strides=2))

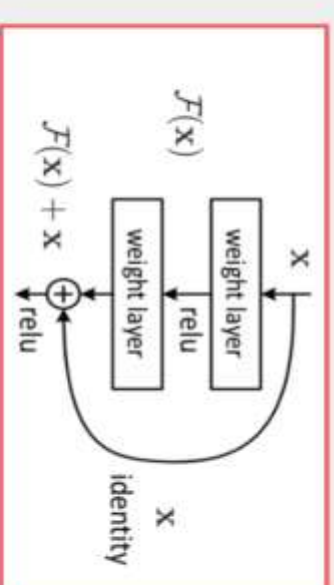
model.add(Conv2D(256,(5,5),padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(3,3),strides=2))

model.add(Conv2D(384,(3,3),padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(384,(3,3),padding='same'))
model.add(Activation('relu'))
model.add(Conv2D(256,(3,3),padding='same'))
model.add(Activation('relu'))
model.add(MaxPooling2D(pool_size=(3,3),strides=2))

model.add(Flatten())
model.add(Dense(units=4096,activation = 'relu'))
model.add(Dense(units=4096,activation = 'relu'))
model.add(Dense(units=23,activation = 'softmax'))
```

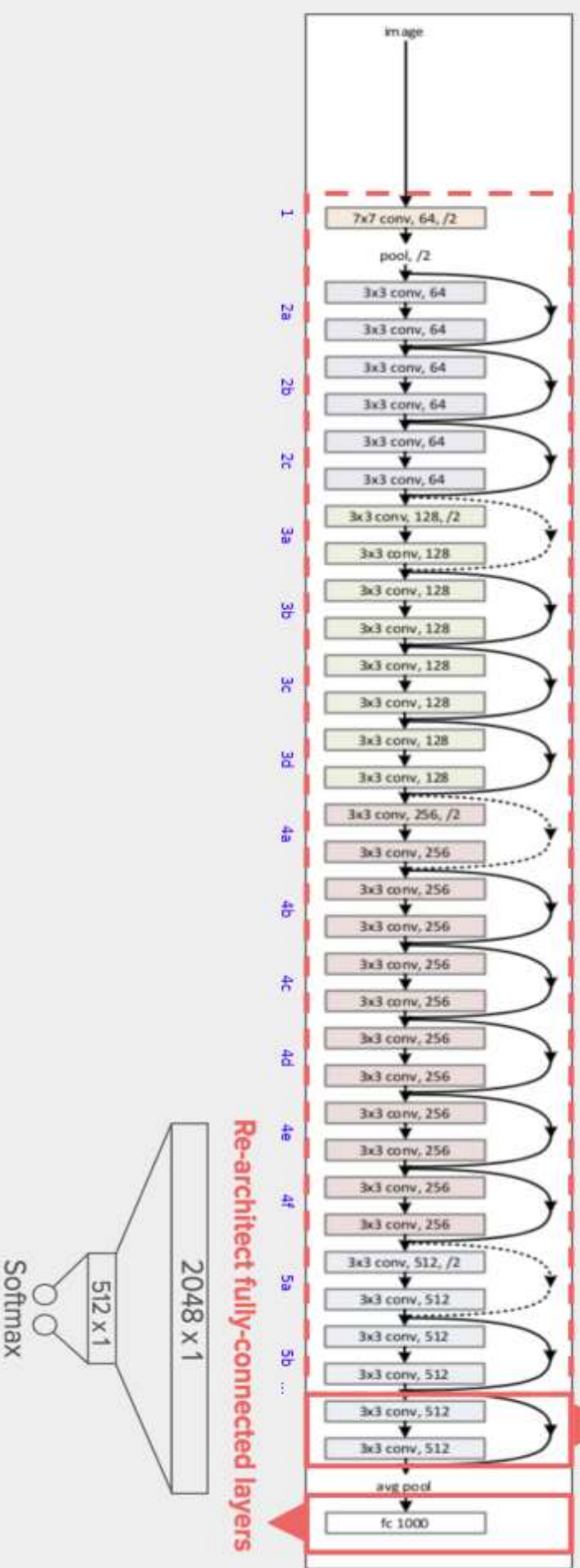
2. Both the networks Resnet50 and Resnet101 did not gave the better result. Accuracy of Resnet50 and Resnet101 were 40% and 47% respectively.

Retrain ResNet50



Residual Learning Block

ResNet50 Diagram



3. Final model architecture and accuracy. This is also a Conv net architecture implemented with batch normalization and accuracy is around 65-70%.

```
model=Sequential()

model.add(Conv2D(96,(11,11),strides=4,input_shape=(227,227,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(3,3),strides=2))

model.add(Conv2D(32,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(Conv2D(32,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)),strides=2))

model.add(Conv2D(64,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(Conv2D(64,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(128,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(Conv2D(128,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))

model.add(Conv2D(256,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(Conv2D(256,(3,3),padding='same',activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))

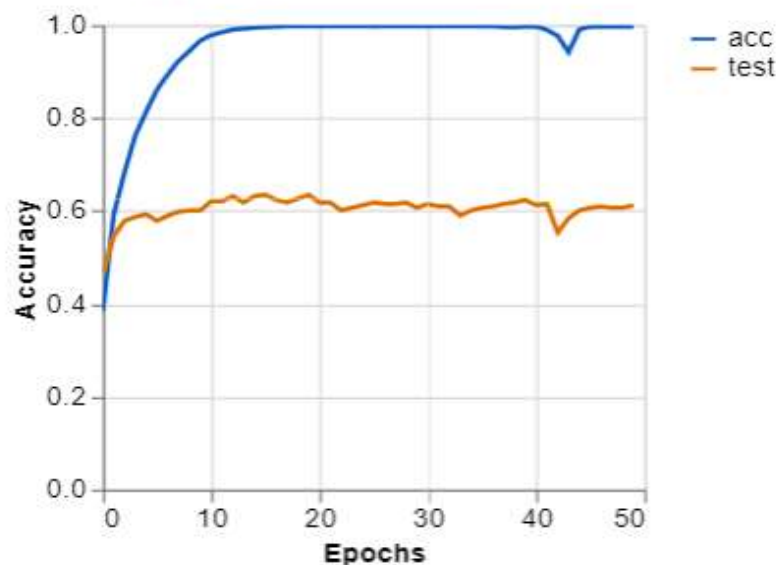
model.add(Flatten())
model.add(Dense(64,activation='relu'))
model.add(BatchNormalization())
model.add(Dense(64,activation='relu'))
model.add(BatchNormalization())

model.add(Dense(units=18,activation = 'softmax'))
```

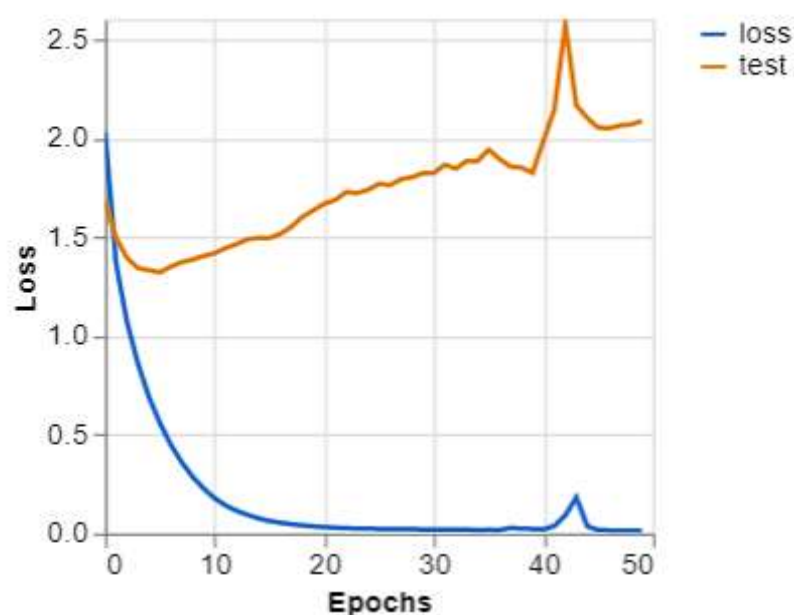
Accuracy And Loss Chart

Blue line showing accuracy and loss per epoch during training where orange line represents the accuracy and loss per epoch during test.

Accuracy per epoch



Loss per epoch



Deploying Our Trained model

After completion of training of our model and we have saved it using Keras API. So now we can use that saved model anywhere like in Android application, Computer Software, Website, etc.

We have chosen Website to develop our project because:-

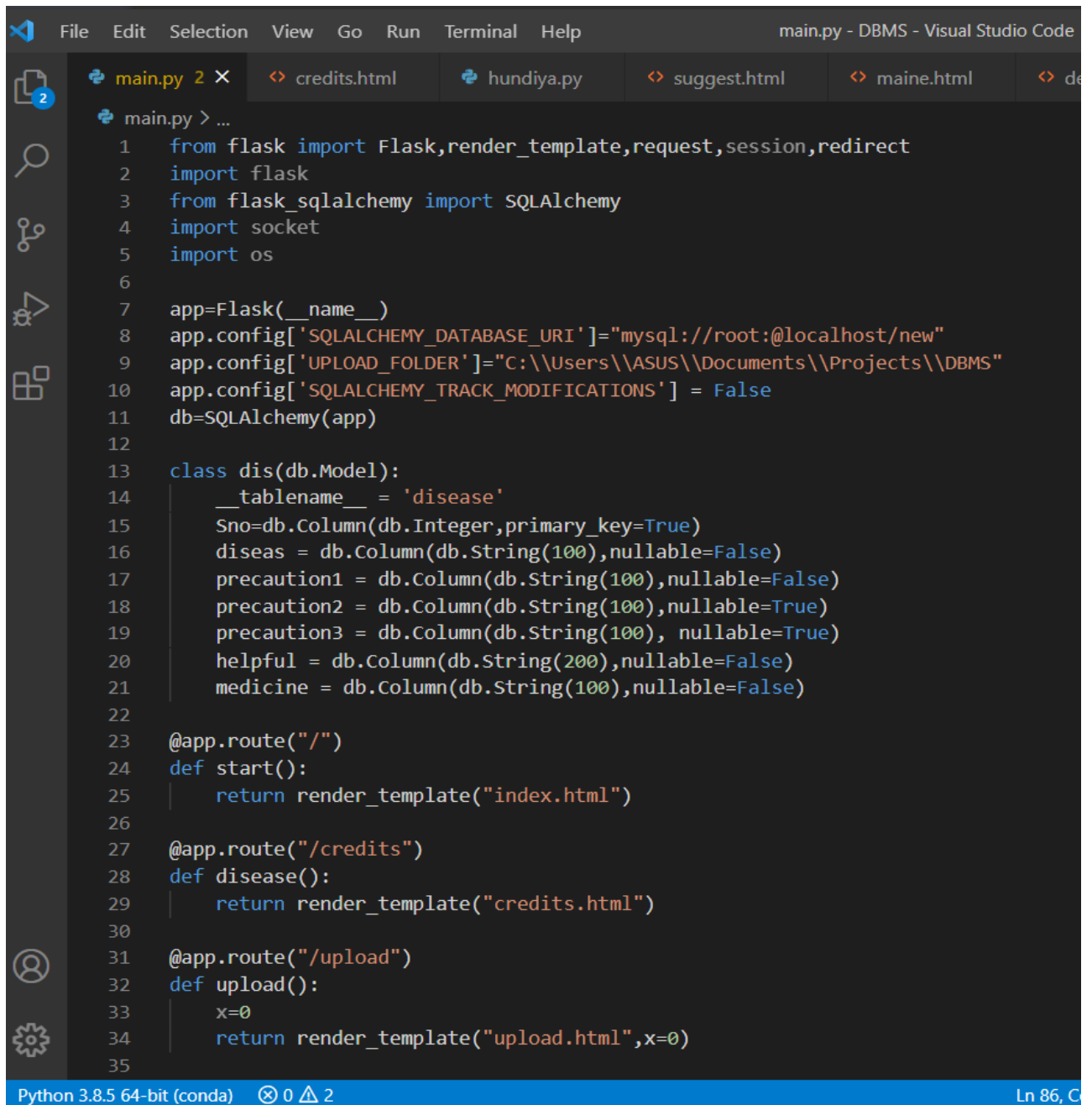
- To provide it free of cost and for better reach to people.
- To save time of user.
- No headache like installation and no space requirement.

Python - Flask

Making this website with flask. Flask is a Python framework or module for backend development and for web applications.



Flask



```
File Edit Selection View Go Run Terminal Help main.py - DBMS - Visual Studio Code

main.py 2 X credits.html hundiya.py suggest.html maine.html de

main.py > ...
1 from flask import Flask,render_template,request,session,redirect
2 import flask
3 from flask_sqlalchemy import SQLAlchemy
4 import socket
5 import os
6
7 app=Flask(__name__)
8 app.config['SQLALCHEMY_DATABASE_URI']="mysql://root:@localhost/new"
9 app.config['UPLOAD_FOLDER']="C:\\Users\\ASUS\\Documents\\Projects\\DBMS"
10 app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
11 db=SQLAlchemy(app)
12
13 class dis(db.Model):
14     __tablename__ = 'disease'
15     Sno=db.Column(db.Integer,primary_key=True)
16     diseases = db.Column(db.String(100),nullable=False)
17     precaution1 = db.Column(db.String(100),nullable=False)
18     precaution2 = db.Column(db.String(100),nullable=True)
19     precaution3 = db.Column(db.String(100), nullable=True)
20     helpful = db.Column(db.String(200),nullable=False)
21     medicine = db.Column(db.String(100),nullable=False)
22
23 @app.route("/")
24 def start():
25     return render_template("index.html")
26
27 @app.route("/credits")
28 def disease():
29     return render_template("credits.html")
30
31 @app.route("/upload")
32 def upload():
33     x=0
34     return render_template("upload.html",x=0)
35

Python 3.8.5 64-bit (conda) 0 2 Ln 86, C
```

Additional Detail Database

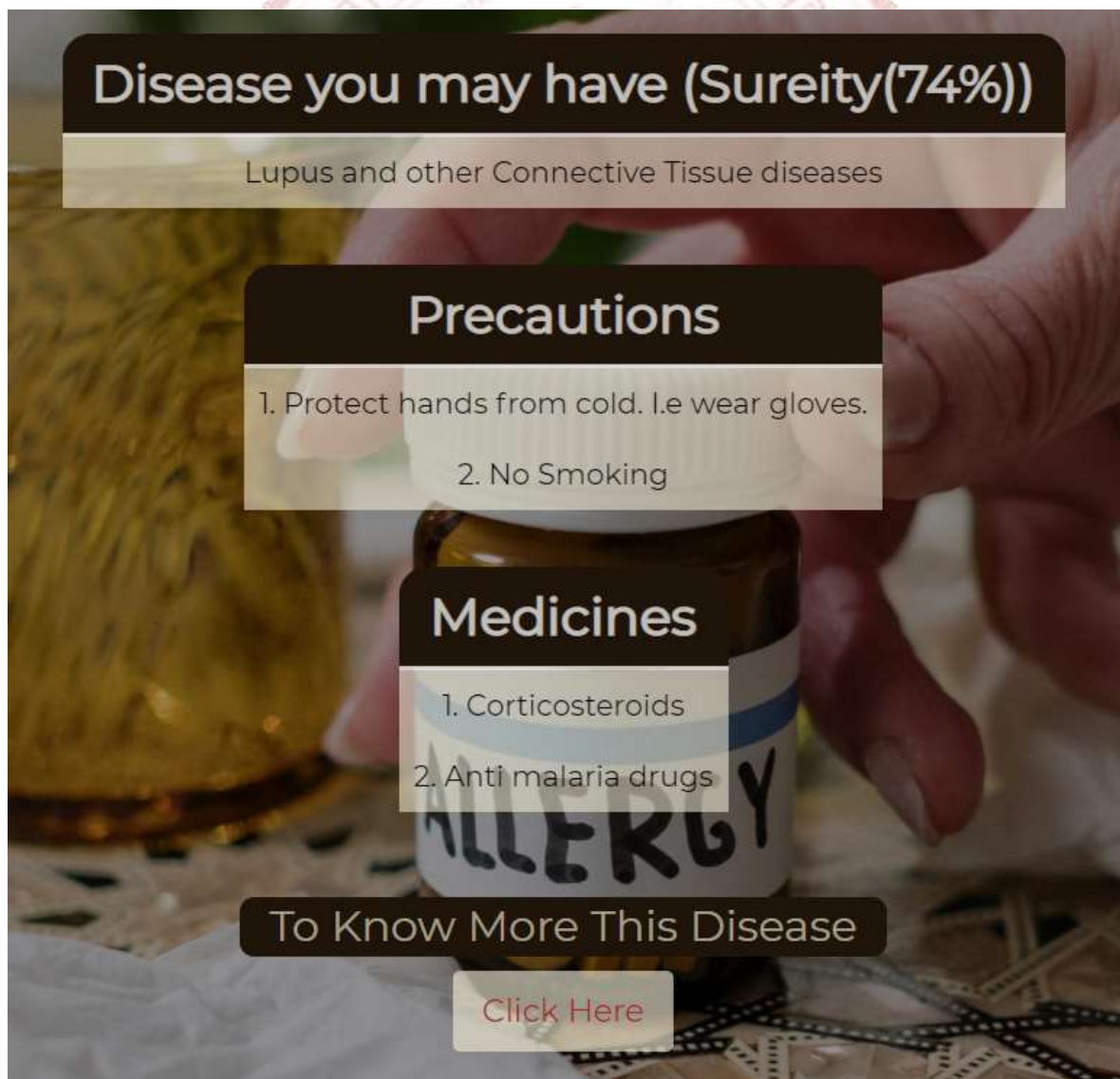
This is the prediction page which is showing the **disease name** and how much percent our model is sure.

Precautions about the disease predicted by the model.

Medicines that are required to stop the disease then and there.

Another **link** is provided for more information

Final Interface



CONCLUSION

We are working with this dataset from last 3 months, we have faced a lot of problem in working with this. While data cleansing was going on there were a lot of disgusting and waste pictures.

We have done each process in detailed from data to the prediction. One more problem is there due to which the accuracy of the model is not so good, Some pictures carries extra things in there.

let me explain this with an example if the problem in teeth, then we need to show our teeth to the dentist not the whole face.

Exactly this dataset consists some picture in which the allergy is at some small space but the picture of whole body is there.

In our vision the solution for this problem is to train a detection model and before applying classification model.

we need to apply detection algorithm which can specify the area of allergy and then classification model will classify the allergic reaction.

This is all what we have done in last few months and our observation is may not be so well but we have tried our best to make this successful. A special thanks to my teacher Mr. Manoj Sethi from our side.

REFERENCES

(1). Automated Skin Disease Identification using Deep Learning Algorithm

Paper by :- Sourav Kumar Patnaik, Mansher Singh Sidhu, Yaagyanika Gehlot, Bhairvi Sharma and P Muthu

Available :- “ <https://biomedpharmajournal.org/vol11no3/automated-skin-disease-identification-using-deep-learning-algorithm/> ”

(2). Network In Network

Paper by :- iuxiang Gu , Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, Xingxing Wang, Li Wang, Gang Wang, Jianfei Cai, Tsuhan Chen

Available :- “<https://arxiv.org/pdf/1512.07108.pdf>”

(3). Convolutional Neural Network

Paper by :- Min Lin, Qiang Chen, Shuicheng Yan

Available :- “<https://arxiv.org/abs/1312.4400>”

DATASET

Available :- “<https://www.kaggle.com/shubhamgoel27/dermnet>”