Husknet:A Real Time Skin Disease Prediction Using Machine Learning.

Rithika M and Rupesh L

Rajalakshmi Engineering College,Chennai

I. Abstract:

Skin diseases are one of the significant global health burdens ,affecting millions of people around the world. Early and accurate diagnosis plays a major role in effective treatment and cure .With advancements in machine learning and medical imaging technologies,predictive modeling has emerged a promising approach for assisting dermatologists in the diagnosis of diseases or infections related to skin .The dataset mainly focuses on the skin diseases images in which there are seven (7) types of diseases that have been used in this paper. The topic of skin disease prediction is explored in relation to a number of machine learning algorithms such as support vector machine,neural networks,decision trees and ensemble approaches out of these Convolutional neural network (CNN) has been chosen as the best option for the training process because it produces a high percentage of accuracy. Results are measured in the means of the accuracy of the image classification in percentage. The Accuracy percentage of the skin diseases predicted in this model is around 97%. Tensorflow, a framework developed by Google offers many tools and resources for building a Convolutional Neural Network(CNN). The integration of AI in dermatological practice creates an impact on patient care by enabling fast and accurate diagnosis of the disease,facilitating early intervention and optimizing treatment strategies that must be used.

Keywords: Convolutional Neural Networks(CNN),Tensorflow,Image Classification

II. Introduction:

Machine learning techniques are widely used in image processing tasks because of their ability to automatically learn the patterns of images and features from vast datasets some of the key uses of machine learning are image classification using Convolutional Neural Network(CNN)[1], object detection using RNN and YOLO (You Only Look Once).

Machine Learning is not only used in picture categorization but also for problems related to object detection .Object detection is the process of identifying things in an image .For example predicting an obstacle present in front of a vehicle. For Object detection YOLO(You only Look Once) and Recurrent Neural network(RNN's) are most commonly used Because both YOLO (You only Look Once) and Recurrent Neural network(RNN's) offer great quickness and precision in real time[2].

One Branch of machine learning is deep learning[3]. It focuses on multi-layered artificial neural networks . Deep learning is used in many places like speech recognition,natural language processing because deep learning models can understand complicated representations of inputs' Deep learning models give great accuracy for Convolution Neural networks(CNN) and Recurrent neural networks(RNN)

Tensorflow is an open source machine learning framework developed by Google[4], [5]. Building Neural networks is made easier and faster by Keras[4], [5] .It makes the user to create models without knowing the underlying math functions inside it for example one can create a Linear Regression model just by importing keras layers instead of writing the entire code by themselves

A popular Python machine learning framework called scikit-learn is used in this paper[6] .scikit-learn offers a wide range of functions for clustering regression and classification .It is a great framework for both beginner and experienced machine learning professionals because it contains comprehensive documentation and an active set of developers to roll out new updates and fix old bugs

In this paper we use Python as the programming language and a Convolutional Neural Network(CNN) based on tensorflow for image classification. This paper uses thousands of photos as input data. At last we compare the accuracy percentage of the model.

III. LITERATURE SURVEY:

In [1] The reference discusses the basics of Convolutional Neural Network (CNN) in python by introducing us to deep learning and artificial neural networks in a simple way and easy to understand. This book focuses on how to use Convolutional Neural Network(CNN) to create powerful image classification models and also take a deeper look into several keras layers used to build Convolutional neural networks (CNN's) Then it also discusses about various activation functions which will eventually lead us to create accurate models which has the ability to perform great task results on various image classification tasks

In [2] [3] discusses about the deep learning techniques,particularly neural networks with many layers to solve tasks for understanding and interpreting data .The goal of deep learning is to develop models and algorithms that can automatically fetch meaningful information from the visual data which enables the machines to perform image classification,image segmentation,object detection and more.This book also provides code examples in python using deep learning libraries such as tensorflow and keras.

In [4] discusses how tensorflow and keras framework build neural network models On CPU And GPU .Tensorflow is the most popular library in numerical computation built from the distributed, mobile and cloud environment. In Tensorflow the data is represented as tensors and the computation is represented as graphs . This book helps us to learn advanced features of tensorflow .

In [5] discusses about use of keras in deep learning to develop smarter and efficient data models This book tells us about the use of keras for different neural networks for training and fine tuning neural network models.This also explains about how to tackle different problems encountered while training different deep learning models.This book contains all details from installation till the execution of the keras framework in the Project.

In [6] discusses the basics of machine learning with scikit library . This tells us about a wide variety of supervised and unsupervised learning methods. These contain many example codes for the aspiring machine learning enthusiasts to learn and experiment with machine learning algorithms. Some basic knowledge of object oriented programming is needed to learn this library.

In [7] discusses about how to build web applications with flask Which is a lightweight python framework to integrate frontend with the backend server to serve requests .It is also discusses how to integrate database services into the web applications and it is the most sought framework for neural network based web apps as the mostly use python for its training and prediction .

Research No	**Name/Year**	**Title**	**Method used**
Research 1	Frank Millstein, 2020.	Convolutional Neural Networks In Python: Beginner's Guide To Convolutional Neural Networks In Python.	Convolutional Neural Networks
Research 2,3	J. Brownlee, 2019	Deep Learning for Computer Vision:Image Classification	Deep Learning
Research 4	A. Fandango,2018	Mastering TensorFlow 1.x	Deep Learning using Tensorflow
Research 5	R. Dua and M. S. Ghotra,2018	Keras Deep Learning Cookbook	Keras basics
Research 6	D. Paper,2019	Hands-on Scikit-Learn for Machine Learning Applications	Learning of basic machine learning algorithms
Research 7	M. Grinberg,2018	Flask Web Development	Developing Web Applications with Python

IV. METHODS:

Based on Figure1 it is a framework of image classification where Convolutional Neural Networks(CNN's) are also applied. This process includes four phases and each phases are

- Install tensorflow with python
- Install flask
- Collect training images
- Train Model using Convolutional neural network
- Classify into Categories

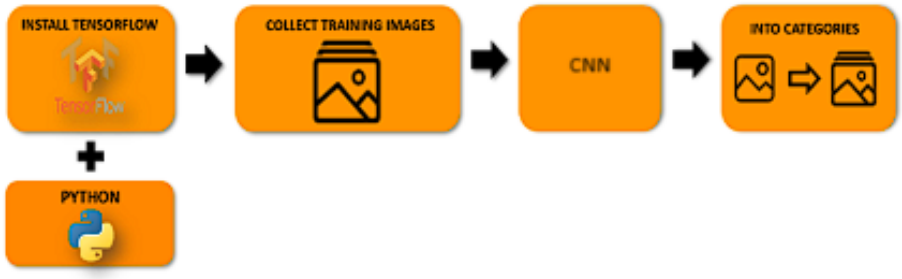


Fig. 1. The block diagram of Image Classification

III.I TRAINING IMAGES

This paper's dataset primarily consists of thousands of images related to skin various skin diseases. These pictures are all from kaggle which is an online community and data science engineers where they can find the datasets that they want to use in the project and can also publish their datasets. In this project the dataset is taken from kaggle which is shown in Figure 1

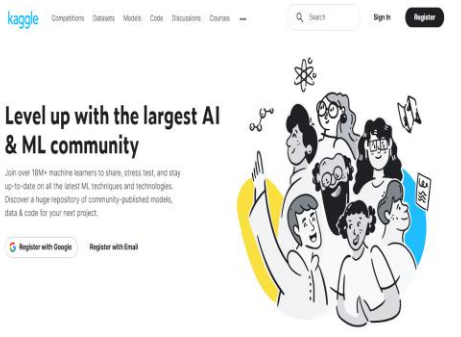


Figure 1.1

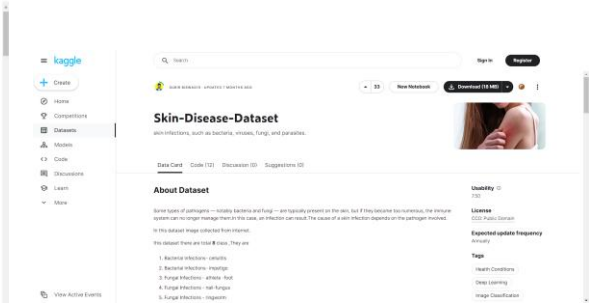
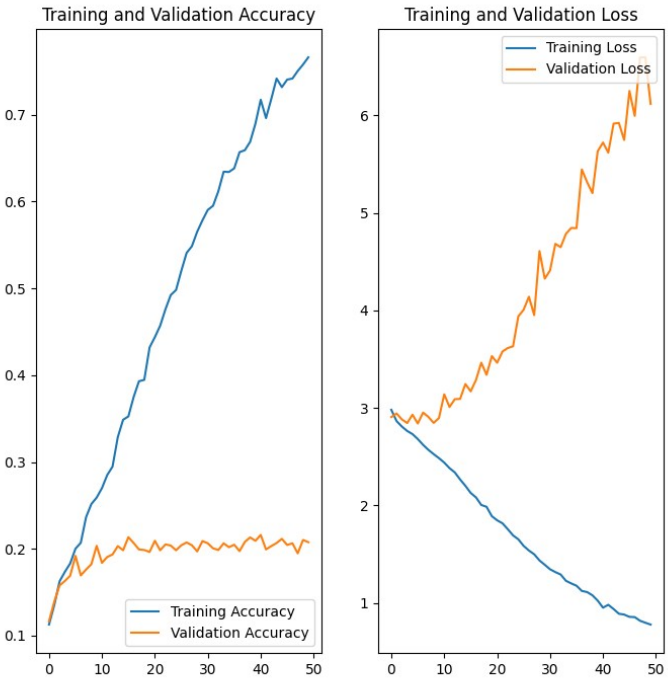


Figure 1.2

This dataset contains two folders which are test images and training images.The training images are of 80% and testing images are of 20%.

No.	**Types of diseases**	**No of images**
1	Cellulitis	136
2	Impetigo	80
3	Nail-fungus	129
4	Ringworm	90
5	Cutaneous-larva-migrans	100
6	Chickenpox	136
7	Shingles	130
Total No of images		**801**



Training and Validation Accuracy:

The y-axis represents the accuracy, measured as a percentage (%).
The x-axis represents the training epochs (iterations over the training data).
The two lines on the graph correspond to:

- **Training Accuracy (blue line):** This reflects how well the model performs on the training data it's being trained on. Ideally, this value should increase steadily as the model learns from the training examples.
- **Validation Accuracy (orange line):** This indicates how well the model performs on a separate set of data (validation set) that the model hasn't been trained on. It's used to assess how well the model generalizes to unseen data and helps prevent overfitting.

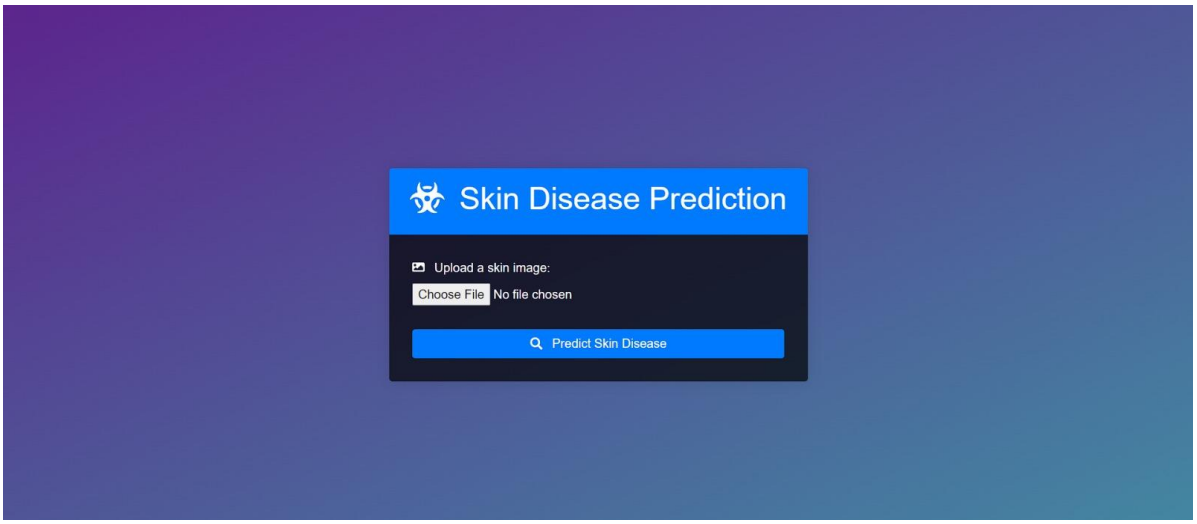
Training and Validation Loss:

The y-axis represents the loss value (typically measured using a loss function like mean squared error or cross-entropy). Lower loss values indicate better model performance.
The x-axis represents the training epochs.
The two lines on the graph correspond to:

- **Training Loss (blue line):** This signifies the model's loss on the training data. It typically decreases as the model learns and reduces its errors on the training examples.
- **Validation Loss (orange line):** This represents the model's loss on the validation data. Ideally, it should also decrease along with the training loss, but not as significantly. A large gap between training loss and validation loss suggests overfitting

In the ideal scenario, both the training and validation accuracy curves should increase over time, and both the training and validation loss curves should decrease over time. A large gap between training accuracy and validation accuracy, or between training loss and validation loss, can indicate overfitting. Overfitting occurs when the model memorizes the training data too well and fails to generalize well to unseen data.

RESULT:



This is the homepage of the skin disease prediction system. It contains a form which is used to send the image file to the backend to load the image into the model.

The form contains a preview box to check what image we have upload before sending it to the backend application.

This is the second page which contains the details about the disease it also contains additional details like cure and disease description.

FUTURE ENHANCEMENT:

To enhance the effectiveness and applicability of our real-time skin disease prediction system, several future updates are planned. These updates aim to improve the accuracy, usability, and integration capabilities of the system, ensuring it remains at the forefront of medical imaging technology. The following areas will be the focus of future enhancements:

Expansion of Dataset: Inclusion of More Diseases: Currently, the dataset includes seven types of skin diseases. Future updates will incorporate a wider variety of skin conditions, increasing the system's diagnostic capability.

Augmentation Techniques: Employ advanced data augmentation techniques to simulate a larger dataset from the existing images, improving the robustness and generalizability of the model.
Model Enhancements:

Algorithm Optimization: Further optimize the Convolutional Neural Network (CNN) architecture to enhance accuracy and reduce computational load.

Incorporation of Advanced Models: Explore the use of more advanced deep learning models like EfficientNet or Vision Transformers (ViTs) which have shown promising results in image classification tasks.

Real-Time Performance Improvement:

Edge Computing Integration: Implement edge computing solutions to perform predictions directly on devices, reducing latency and improving real-time performance.

Model Quantization: Use model quantization techniques to reduce the model size and speed up inference times without significantly sacrificing accuracy.

User Interface and Experience:

Enhanced User Feedback: Develop a more interactive and informative user interface that provides detailed feedback on predictions, including confidence scores and visual explanations (e.g., Grad-CAM).

Mobile Application Development: Create a mobile application version of the system, making it more accessible for on-the-go diagnostics.

Interoperability and Integration:

Integration with Electronic Health Records (EHR): Enable seamless integration with existing EHR systems to streamline the workflow for dermatologists and other healthcare providers.

API Development: Provide an API for easy integration with other healthcare applications and platforms, facilitating broader adoption.

Continuous Learning:

Active Learning Framework: Implement an active learning framework where the model can be continuously updated with new data, improving its performance over time.

User-Generated Data: Allow healthcare providers to contribute anonymized data from real-world use to further refine and validate the model.

Regulatory and Ethical Compliance:

Compliance with Medical Standards: Ensure that the system meets all relevant regulatory standards and guidelines for medical devices, enhancing its credibility and trustworthiness.
Ethical AI Practices: Adhere to ethical AI practices, ensuring patient data privacy and unbiased model predictions.

Extended Use Cases:

Teledermatology Support: Enhance features to support teledermatology, enabling remote consultations and diagnosis.
Multilingual Support: Add support for multiple languages to cater to a broader audience and improve accessibility.

REFERENCES:

[1] F. Millstein, *Convolutional Neural Networks In Python: Beginner's Guide To Convolutional Neural Networks In Python*. Frank Millstein, 2020.

[2] J. Brownlee, *Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*. Machine Learning Mastery, 2019.

[3] K. Saitoh, *Deep Learning from the Basics: Python and Deep Learning: Theory and Implementation*. Packt Publishing Ltd, 2021.

[4] A. Fandango, *Mastering TensorFlow 1.x: Advanced machine learning and deep learning concepts using TensorFlow 1.x and Keras*. Packt Publishing Ltd, 2018.

[5] R. Dua and M. S. Ghotra, *Keras Deep Learning Cookbook: Over 30 recipes for implementing deep neural networks in Python*. Packt Publishing Ltd, 2018.

[6] D. Paper, *Hands-on Scikit-Learn for Machine Learning Applications: Data Science Fundamentals with Python*. Apress, 2019.

[7] M. Grinberg, *Flask Web Development: Developing Web Applications with Python*. "O'Reilly Media, Inc.," 2018.