North East University Bangladesh

Department of Computer Science and Engineering

“X-ray and Eye Disease Image Prediction System”

By

Md. Abdul Mutalib
Reg. No: 190303020001
BSc(Engg) in CSE
4th year 2nd semester

Shahriar Hussain
Reg. No: 190303020008
BSc(Engg) in CSE
4th year 2nd semester

Kopil Das
Reg. No: 190303020009
BSc(Engg) in CSE
4th year 2nd semester

Supervised By

Tasnim Zahan
Assistant Professor
Department of Computer Science and Engineering
North East University Bangladesh

25th January, 2024

“X-ray and Eye Disease Image Prediction System”

A Thesis submitted to the Department of Computer Science and Engineering,
North East University Bangladesh, in partial fulfillment of the requirements
for the degree of Bachelor of Science in Computer Science and Engineering

By

Md. Abdul Mutalib
Reg. No: 190303020001
BSc(Engg) in CSE
4th year 2nd semester

Shahriar Hussain
Reg. No: 190303020008
BSc(Engg) in CSE
4th year 2nd semester

Kopil Das
Reg. No: 190303020009
BSc(Engg) in CSE
4th year 2nd semester

Supervised By

Tasnim Zahan
Assistant Professor
Department of Computer Science and Engineering
North East University Bangladesh

25th January, 2024

Recommendation Letter from Thesis Supervisor

These Students, *Md. Abdul Mutalib, Shahriar Hussain, Kopil Das*, whose thesis/project entitled "*X-ray and Eye Disease Image Prediction System*", is under my supervision and agrees to submit for examination.

Signature of the Supervisor :

Tasnim Zahan
Assistant Professor
Department of Computer Science and Engineering
North East University Bangladesh

Qualification Form of BSc (Engg) Degree

Student Name : Md. Abdul Mutalib, Shahriar Hussain, Kopil Das
Thesis Title : X-ray and Eye Disease Image Prediction System

This is to certify that the thesis is submitted by the student named above in July, 20023. It is qualified and approved by the following persons and committee.

Head of the Dept.

Mr. Rathindra Chandra Gope
Assistant Professor
Department of CSE
North East University Bangladesh

Supervisor

Tasnim Zahan
Assistant Professor
Department of CSE
North East University Bangladesh

Abstract

This final-year project explores the application of machine learning techniques to address critical healthcare challenges. The project focuses on three key objectives: pneumonia classification, eye disease classification, and the development of a user-friendly frontend for image-based predictions. To achieve these goals, transfer learning, convolutional neural networks (CNN), and a naive Bayes classifier were employed. The development of the frontend provides a seamless interface for users to submit medical images and obtain real-time predictions for pneumonia and eye diseases. The project not only showcases the effectiveness of machine learning in medical diagnostics but also highlights the significance of user-friendly interfaces in healthcare applications.

Keywords: Pneumonia, Machine Learning, Chest X-ray images, Naive Bayes classifier, Support Vector Machine, Transfer Learning, DenseNet161, Feature Extraction, Data Augmentation

Table of Contents

List of Figures

Chapter 1

INTRODUCTION

Introduction: Pneumonia is a leading cause of death in Bangladesh, accounting for 13% of deaths in children under the age of five. In 2021, an estimated 12,000 children in Bangladesh died from pneumonia. This is a major public health problem, and it is exacerbated by the fact that Bangladesh has a shortage of radiologists which is not enough to meet the needs of the population.

Machine learning (ML) can be used to help diagnose pneumonia. ML algorithms can be trained on a dataset of chest X-ray images to learn to identify the telltale signs of pneumonia. This can help to improve the accuracy of diagnosis, and it can also help to reduce the workload on radiologists.

In this project, we explore different ML implementations that can be used to classify pneumonia from chest X-ray images. We will be looking at a Naive Bayes classifier, a support vector machine (SVM) classifier, a transfer learning-based classifier, and convolutional neural networks (CNN). Our input is in the form of chest X-ray images, and we output a classification of the presence of pneumonia.

In addition, we also explore different dataset into our selected Ml algorithms. We trained on a dataset of eye disease images to predict 4 different class.

On the other-hand, the Streamlit-based application allows users to upload images, facilitating predictions for pneumonia or normal image. Beyond chest X-ray datasets, for trained our models on an eye disease dataset, providing users with a versatile tool to predict various health conditions.

Chapter 2

RELATED WORKS

2.1 Image Feature Extraction

Image feature extraction is a crucial step in computer vision and image processing tasks. It involves transforming raw images into a meaningful and compact representation of their visual characteristics. These extracted features capture essential information from the images and can be used for various purposes, such as image classification, object detection, image retrieval, and more.

There are several popular techniques for image feature extraction, and some commonly used methods include:

2.2 Discrete Cosine Transform (DCT)

In their 2010 paper titled "Bayesian Classification Using DCT Features for Brain Tumor Detection" Quratul Ain,Irfan Mehmood,Muhammad Naqi and Arfan Jaffar Discusses DCT approaches for image feature extraction, which can then be fed as inputs to Naive Bayes for an image classification task. In their peper, they use DCT with various algorithm and got better result. In his 2016 paper titled "Image Classification Using Naive Bayes Classifier" Park identifies DCT as the best feature extraction method since it even beats many of the nueral network classification models he tried. Thus we decided to similarly use the discrete cosine transform to extract features for our Naive Bayes classifier.

2.3 Histogram Oriented Gradient (HOG)

The Histogram of Oriented Gradients (HOG) feature extraction descriptor has been widely studied and applied in image classification tasks. HOG, introduced by Dalal in 2005, is renowned for its ability to capture shape and edge information in images, making it a popular choice for image classification.

HOG operates by dividing the image into small cells and computing histograms of gradient orientations within each cell. These histograms effectively capture the local patterns and textures in the image, forming the foundation of the HOG descriptor. The selection of HOG parameters plays a crucial role in its performance.

Researchers have extensively explored various aspects of HOG to enhance its effectiveness in image classification. The choice of cell size determines the level of detail captured by the descriptor, with smaller cells capturing finer details at the expense of increased computational complexity. The number of orientation bins determines the level of granularity in the gradient computations, allowing the descriptor to capture different edge orientations accurately.

Furthermore, researchers have investigated gradient computation methods and normalization techniques to improve the robustness and discriminative power of HOG features. These advancements have contributed to the success of HOG in accurately classifying images into different categories.

In the realm of image classification, numerous researchers have contributed to the understanding and advancements of HOG. Notably, the work by Bouchra NASSIH*, Aouatif AMINE, Mohammed NGADI, Nabil HMINA [Add complete reference here] has provided valuable insights into the variations and applications of HOG in various image classification scenarios.

2.4 Data Augmentation

We were very concerned about our model's performance, that's why we want to improve generalization and reduce overfitting issue. For this reason, we explore various techniques to enhance the diversity and quantity of image data for training our models. By augmenting the training data, we can expose the model to a wider variety of variations and increase its ability to generalize well to unseen data.

2.5 Normalization

We applied the normalization technique to improve the accuracy of the transfer learning model. Normalizing means getting the mean and standard deviation for each channel. The mean is the average of the data points. It indicates the center of the dataset. The standard deviation is the square root of the variance. A high standard deviation suggests that the values are dispersed throughout a larger range, whereas a low standard deviation suggests that the values tend to be near the predetermined mean.

We got the mean value [0.0020, 0.0020, 0.0020] and the standard deviation value [0.0010, 0.0010, 0.0010] from our input dataset. After that, we got 88.94% accuracy while using these values. which means the test accuracy has improved compared to the previous test accuracy, which was 87.50%.

2.6 Architecture Modification

It's an experiment with modifying the architecture slightly. For example, we can add additional convolutional layers. So, in our task, we already have 161 layers, and we tried to add an extra 3 additional layers.

2.7 Class imbalance handling

Class imbalance occurs when one class significantly outnumbers the other class in the dataset. In our dataset, we have two classes.

1. 1341 images in the normal class
2. 3875 images in the pneumonia class.

so, we can easily claim that our dataset is class imbalanced. Therefore, in this case, we need class imbalance handling. There are two types of class imbalanced handling.

1. Oversampling technique
2. Under-sampling.

In our project, we just used the oversampling technique. So, we are going to discuss it.

This involves randomly duplicating instances from the minority class to balance the class distribution. In order to maintain a balanced class distribution, it selects an instance at random from the minority class. To use the oversampling technique, we can use techniques like random oversampling or the synthetic minority oversampling technique (SMOTE) to generate synthetic samples. Here we used SMOTE to generate synthetic samples.

- How SMOTE works:
    - SMOTE is a technique used to address the class imbalance problem in machine learning, particularly in classification tasks where one class significantly outnumbers the other. SMOTE focuses on the minority class. It works by generating synthetic samples in the feature space, creating new instances that resemble the existing minority class examples. This is done by interpolating between existing minority-class instances.
- Algorithm steps:
    - For each minority class instance, SMOTE selects the k nearest neighbors (commonly, k = 5).
    - Synthetic instances are created by selecting one of the k neighbors and generating a synthetic sample along the line segment joining the minority instance and its selected neighbor.
    - This process is repeated until the desired level of oversampling is achieved.

2.8 Data Set

We partitioned the data into training, validation, and testing sets using Kaggle's X-ray scan datasets. There were 5856 images, which led to a split of 5216, 16, and 624. In contrast, we gathered our dataset on eye diseases from Kaggle around 4217 images and divided it into three categories: training (3372), validation (419), and testing (426). and divided it into three categories: training, validation, and testing (80%, 10%, and 10%, respectively).

Data augmentation is a method of applying distortion to the original dataset. It is used to increase the size and diversity of a training dataset. It applies various transformations to the existing data samples. The goal of data augmentation is to enhance the model's ability to generalize and improve its performance. In our data augmentation section, we used different transformations like Random flip, random rotation, random translation, gray scaling and random crop etc. Here we explained our transformations briefly.

- Gray scale: Grayscale conversion is one such transformation that can be applied to images. It is a representation of an image where each pixel's intensity is represented by a single value, ranging from 0 (black) to 255 (white). Gray scale conversion takes a color image, which typically consists of three-color channels (red, green, blue), and reduces it to a single channel representation.
- Random flip: Random horizontal flip is one such transformation that can be applied to images. Where an image is flipped horizontally with a certain probability. This transformation involves reversing the image from left to right, simulating a mirror reflection.
- Random rotation: This technique involves randomly rotating images by a certain angle within a specified range. By randomly rotating images, the model learns to recognize objects and patterns from different angles, making it more versatile in handling images with various orientations.
- Random translation: This transformation involves randomly shifting or translating images in both the horizontal and vertical directions by a certain number of pixels. By randomly shifting images, the model learns to recognize objects and patterns regardless of their specific position.
- Random crop: This transformation involves randomly selecting and cropping a portion of an image. The size of the crop is typically specified as a range or maximum dimensions in pixels.

Here are the examples of our augmented images:

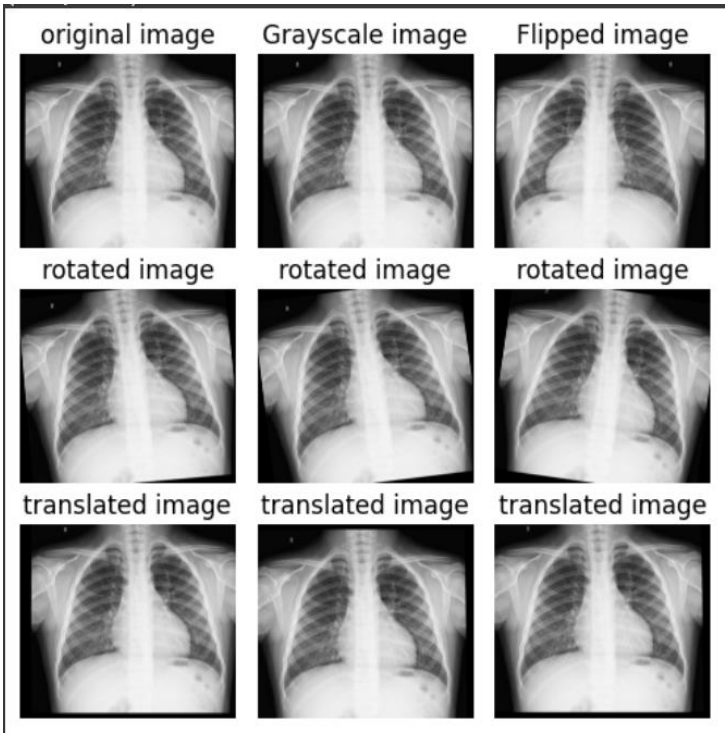

Figure 1: Augmented data for Transfer Learning

Chapter 3

PROPOSED METHOD

3.1 Support Vector Machine (SVM)

The Model: Support Vector Machines (SVM) are a powerful and widely used machine learning algorithm for classification tasks. They belong to the family of supervised learning models and have gained popularity due to their effectiveness and versatility in handling various types of data. SVM can be used for both binary and multi-class classification problems.

One of the key characteristics of SVM is the ability to find an optimal hyperplane that separates different classes in the feature space. This hyperplane is chosen to maximize the margin, which is the distance between the hyperplane and the nearest data points of each class. By maximizing the margin, SVM aims to achieve better generalization and improve the model's ability to classify new, unseen data accurately.

SVM can handle linearly separable data, where a straight line or hyperplane can separate the classes. However, they can also handle non-linearly separable data by using kernel functions. Kernel functions transform the original feature space into a higher-dimensional space, where the data becomes linearly separable. This allows SVM to capture complex relationships and patterns in the data.

How SVM works: Support Vector Machines (SVM) is a classification algorithm that aims to find an optimal line or hyperplane that can separate data points belonging to different classes. The key idea behind SVM is to maximize the margin, which is the distance between the separating line/hyperplane and the nearest data points of each class. This margin provides a safety buffer, reducing the chances of misclassifying new, unseen data.

SVM can handle cases where the data points are perfectly separable by a straight line or hyperplane. However, it is also designed to handle cases where the data is not linearly separable. In such cases, SVM uses a technique called the "kernel trick." The kernel trick transforms the original feature space into a higher-dimensional space, where the data becomes separable by a linear boundary. This allows SVM to capture complex relationships and patterns in the data.

By finding the optimal separating hyperplane, SVM learns the best decision boundary between classes. The points closest to the hyperplane, called support vectors, play a crucial role in

defining the boundary. SVM focuses on these support vectors and ignores other data points, making it efficient in high-dimensional spaces.

The choice of the kernel function in SVM is critical. Different kernel functions define different relationships between features, allowing SVM to handle various types of data. Common kernel functions include the linear kernel for linear separability, polynomial kernel for non-linear decision boundaries with higher degrees, and radial basis function (RBF) kernel for non-linear and complex patterns.

In our SVM model, we employed the HOG and LBP feature extraction techniques to feed our data to the SVM model. HOG captures shape and edge information by analyzing local gradient orientations, while LBP characterizes texture by examining pixel intensity comparisons. These features provide meaningful representations of the images, enabling the SVM model to learn discriminative patterns and make accurate class predictions.

Method for SVM: In our SVM model we utilized two feature extraction techniques, namely Histogram of Oriented Gradients (HOG) and Local Binary Patterns (LBP), in combination with the Support Vector Machines (SVM) algorithm.

LBP is a texture analysis technique that focuses on the pixel intensity comparisons within a local neighborhood. It operates by comparing the intensity value of a central pixel with its surrounding neighbors and encoding the results as binary patterns. These binary patterns capture the local texture variations in the image. LBP features are computed by constructing histograms of the different binary patterns occurring within the image. These histograms serve as descriptors that represent the texture information in the image. LBP features are known for their simplicity and effectiveness in texture classification tasks.

Local Binary Patterns (LBP) is a texture analysis technique widely used in computer vision for its simplicity and effectiveness in capturing local texture variations in an image. LBP operates by comparing the intensity value of a central pixel with its surrounding neighbors. This comparison is done by thresholding the intensity values, resulting in binary values of 0 and 1.

To compute LBP features, a local neighborhood is defined around each pixel in the image. The neighborhood can be circular or rectangular and can vary in size. For each pixel, the binary pattern is created by comparing the intensity value of the central pixel with its neighboring pixels. If a neighbor's intensity value is greater or equal to the central pixel's value, a binary value of 1 is assigned; otherwise, a binary value of 0 is assigned.

3.2 Naive Bayes Classifier

The model: The Naive Bayes model was chosen as the initial algorithm for our pneumonia detection model due to its simplicity and ability to provide a baseline accuracy. This allowed us to gain insights into the dataset and establish a starting point for further improvements with more complex models.

How it works: The Naive Bayes model works by making the Naive Bayes assumption, which assumes that the probability of each feature occurring is conditionally independent of all other features given the class. While this assumption is not entirely accurate, it offers a practical simplification that often performs well in practice. The simplification allows a computation for the probability of some input X belong to come class C to be,

$$P(class = features = X) = rj_i \, P(class = features_i = X_i)$$

This expression can be computed much more easily by counting the number of times feature i occurs in class C vs the total number of times feature i occurs at all.

Image Pre-processing: To ensure standardized input for our Naive Bayes classification model, we performed pre-processing steps on our dataset. we resized all images to a fixed size of 100 x 100 pixels, maintaining consistent dimensions across the dataset. By doing so, we ensure that the images are in a uniform format for further processing and analysis. While augmentation data , we converted all the images to grayscale.

Feature Extraction: As the Naive Bayes model is not capable of directly processing raw pixel data like more complex deep learning models, we incorporated effective feature extraction techniques to derive informative features from the image data. We experimented with only Discrete Cosine Transform (DCT).

DCT is commonly employed in image compression, where images are represented as a weighted sum of cosine waves. By calculating the DCT on a sliding square of pixels, typically 8 x 8, local DCT values are obtained. In our case, the 100 x 100-pixel images were divided into a grid, and DCT was computed for each square. This resulted in a grid of DCT values, capturing frequency components of the image. We performed DCT calculations using different window sizes, including 4 x 4, 8 x 8, to experiment and identify the optimal size for our model.

3.3 Transfer Learning

Transfer learning is a technique where we can take a pre-trained model. Typically trained on a large-scale dataset, and adapting it to a new, smaller dataset or a different task (target task). In our project our target task is identifying whether a chest x-ray image shows signs of pneumonia or normal. When we used transfer learning to solve our target task, we selected a pre-trained model as our base model. There are may two possible approaches to using knowledge from the pre-trained model. The first way is to freeze few layers of the pre-trained model and train other layers on our new dataset for the new task. The second way is to make a new model, but also take out some features from the layers in the pre-trained model and use them in a newly created model. For our target task we have selected the first approach. We will discuss about our approach in our classification layer section.

3.3.1 Base model

We have selected DenseNet161 as our pretrained model. DenseNet161 is a deep learning neural network model used for image classification tasks. It is based on a DenseNet architecture, which means that it has densely connected layers where each layer receives input from all preceding layers. DenseNet architecture have different model such as DenseNet121, DenseNet161, DenseNet169 and so on. DenseNet161 has 161 layers. It was introduced in a research paper in 2017 and has achieved state-of-the-art results on several image classification tasks, including the ImageNet dataset. Its architecture includes convolutional layers, pooling layers, and fully connected layers. It also uses techniques such as batch normalization and dropout to improve its performance. Now we are going to discuss briefly different types of layers in DenseNet architecture.

- Convolutional layers: Convolutional layers are the fundamental building blocks of DenseNet-161. They perform convolution operations on the input data to extract features and capture spatial patterns. In DenseNet-161, multiple convolutional layers are stacked together to form dense blocks.
- Bottleneck layers: Within each dense block, bottleneck layers are employed. A bottleneck layer consists of a batch normalization layer, a 1x1 convolutional layer, and a rectified linear unit (ReLU) activation function. The 1x1 convolutional layer reduces the number of input channels, often referred to as the bottleneck layer, significantly reducing computational complexity.

- Pooling layers: After the convolutional layers, pooling layers are inserted to down-sample the input image and reduce the size and computational complexity. Two common types of pooling layers are:
    - Max Pooling: it selects the highest pixel value from the window of the image currently covered by the feature detector which helps to identify the salient features of the image.
    - Average Pooling: it calculates the average value from the window of the image currently covered by the feature detector which identifies the full extent of the image.

There are few other terms which are the important concept in DenseNet.

- Growth rate: growth rate determines the number of feature maps are generated into individual layers inside dense blocks.
- Dense connectivity: In DenseNet-161, layers within a dense block are densely connected to each other in a feed-forward manner. Each layer takes as input the feature maps from all preceding layers within the same dense block.
- Composite functions: Composite functions refer to mathematical functions that are formed by combining two or more individual functions. In a composite function, the output of one function becomes the input for another function, creating a chain of transformations. So we can define as a composite function of three consecutive operations they are batch normalization, ReLU and a 3x3 convolution (Conv).

3.3.2 Classification layers

The classification layer in DenseNet architecture is the final layer of the neural network that is responsible for performing the task of image classification. It takes the features by the preceding layers and maps them to the specific output classes. As our dataset is not very large so we have decided to freeze the whole pre-trained model layers and simply add a linear classifier at the end that take the output of DenseNet161 as its input and produces outputs two classes (Pneumonia or Normal). Our DenseNet161 architecture is shown below for Pneumonia and Eye disease.

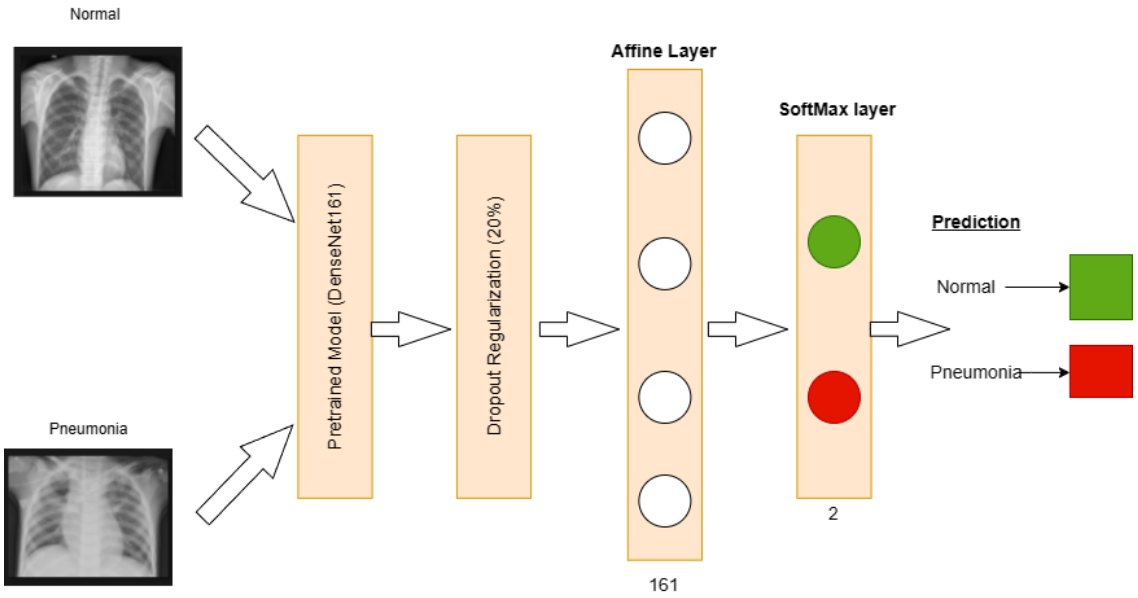

Figure 2 DenseNet161 Architecture for Pneumonia classification

classifying eye disease, whether it is cataract, diabetic retinopathy, glaucoma, or normal. We used transfer learning to classify eye disease and DenseNet161 as a pretrained model. As we used this model in our previous classification task, in this case, we are going to talk about the changes we made in the DenseNet161 architecture for classifying eye disease. Actually, we mainly made changes in the classification layer section and normalization values. The normalization values were mean = [0.0014, 0.0012, 0.0007] and standard deviation = [0.0007, 0.0006, 0.0004] for classifying eye disease.

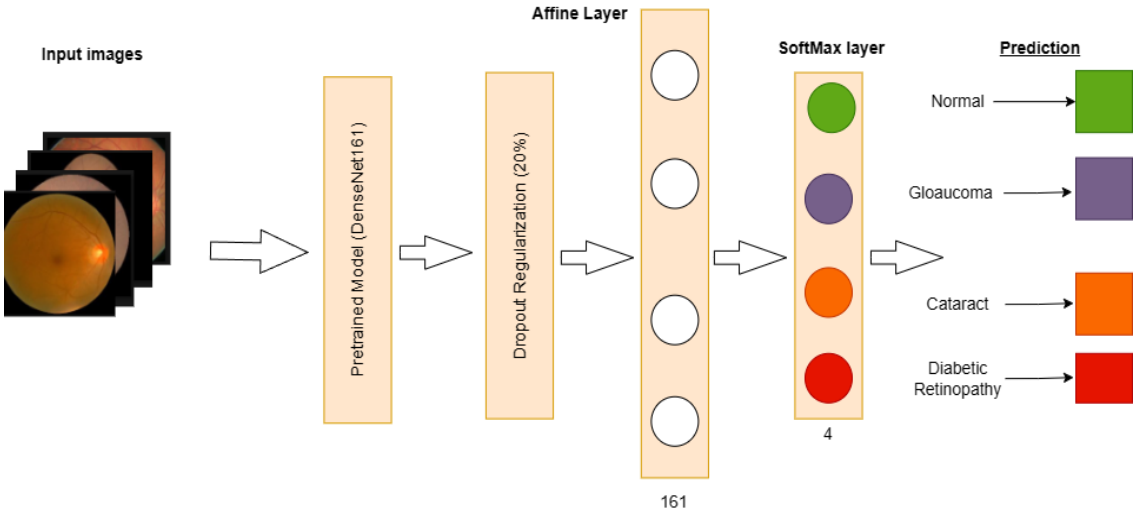

Figure 3 DenseNet161 Architecture for Eye Diseases classification

3.4 Convolutional Neural Network (CNN)

Model Overview:

To broaden the utility of our system, we extended the CNN to predict not only pneumonia in chest X-ray images but also various eye diseases. The CNN architecture, designed for multi-class classification, demonstrates adaptability by handling four distinct eye disease categories: cataract, diabetic retinopathy, glaucoma, and normal.

Evolution of the CNN Model:

The CNN architecture is composed of key layers strategically designed to unravel complex patterns within chest X-ray images. Conv2D layers perform convolution operations, extracting high-level features. Our journey involved experimenting with various architectures, initially implementing a (5x5) Convolution followed by ReLU activation, (3x3) Convolution, ReLU, and finally, an Affine Softmax layer. While this achieved a commendable 85% accuracy, we observed a dip in performance after applying transformations to the training set.

Incorporating MaxPool Layers for Robustness:

To enhance robustness and address translational invariances, we introduced MaxPool layers after every activation function. MaxPooling2D layers reduce spatial dimensions, aiding computational efficiency. Dense layers at the end contribute to the final classification, capturing intricate relationships among features. This adjustment resulted in the refined model: (5x5) Convolution → ReLU → MaxPool → (3x3) Convolution → ReLU → MaxPool → Affine Softmax. Dense layers at the end contribute to the final classification, capturing intricate relationships among features. This modification not only bolstered generalization but also contributed to improved accuracy.

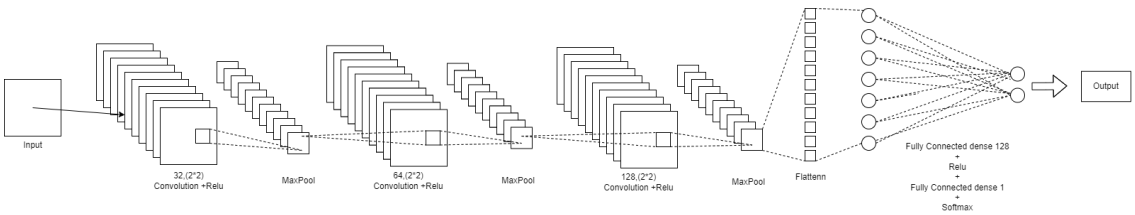

Figure 4: CNN Architecture for Pneumonia classification

Image Size Enhancement and Hyperparameter Tuning:

Recognizing the importance of image size, we increased it from 100x100 to 258x258. This adjustment facilitated the extraction of more expressive features, elevating the model's accuracy in medical image analysis. . During data augmentation, grayscale conversion enhanced the model's ability to discern subtle variations in X-ray images. These pre-processing steps promote uniformity in data representation.

Hyperparameter tuning was a crucial phase. Learning rates ranging from 0.01 to 0.00001 were explored, revealing that a learning rate of 0.001 yielded optimal accuracies with rapid convergence. Iterative testing also determined that two hidden layers with channel sizes 24 and 12 minimized overfitting while maximizing accuracy.

Chest X-ray Image Training:

For pneumonia prediction, the CNN was trained on a dataset comprising chest X-ray images. Employing Conv2D layers with varying filter sizes and pooling layers facilitated the extraction of relevant features. The model underwent rigorous training for 15 epochs, converging to an accuracy of 78.04% on the test set.

Eye Disease Image Training:

Expanding its capabilities, the same CNN was repurposed for eye disease prediction. The model, configured to accommodate four classes, underwent training on a diverse dataset encompassing cataract, diabetic retinopathy, glaucoma, and normal eye images. After 15 epochs, the CNN achieved a commendable accuracy of 65.02% on the test set.

Analysis:

The CNN accuracy history depicted the impact of learning rates on model performance, guiding our decision to limit training to fifteen epochs to mitigate overfitting. The meticulous adjustment of hyperparameters showcased the delicacy in balancing model complexity and performance.

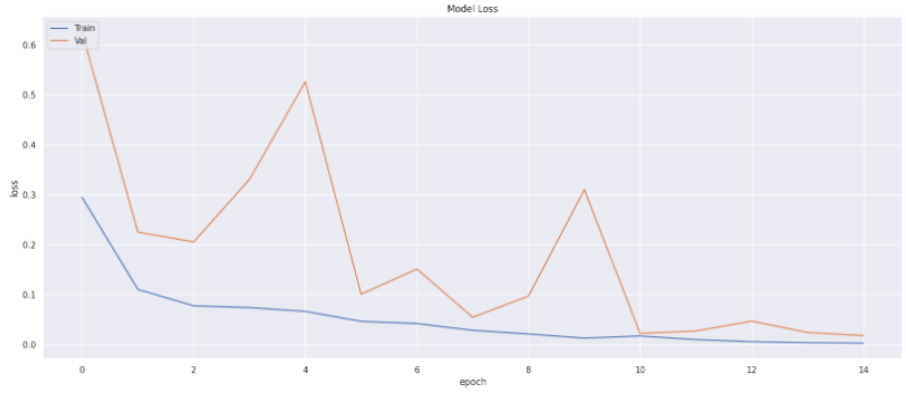

Figure 5: Training and validation loss over every epoch in chest x-ray image

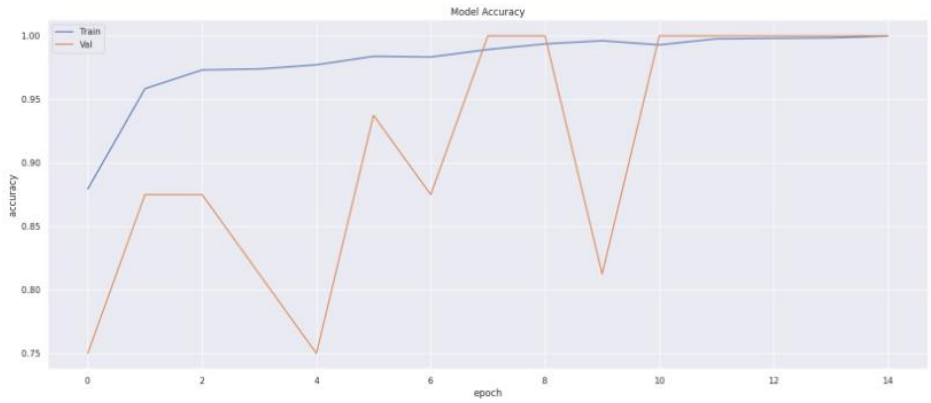

Figure 6: Training and validation accuracy over every epoch in chest x-ray image

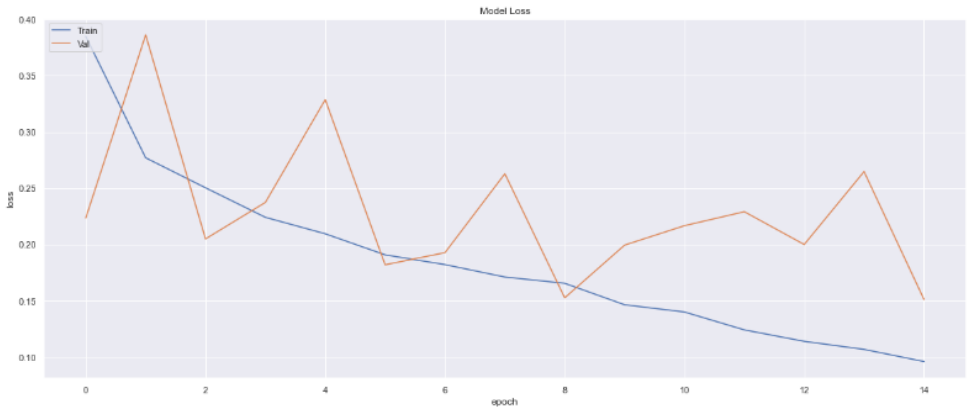

Figure 7: Training and validation loss over every epoch in eye diseases image

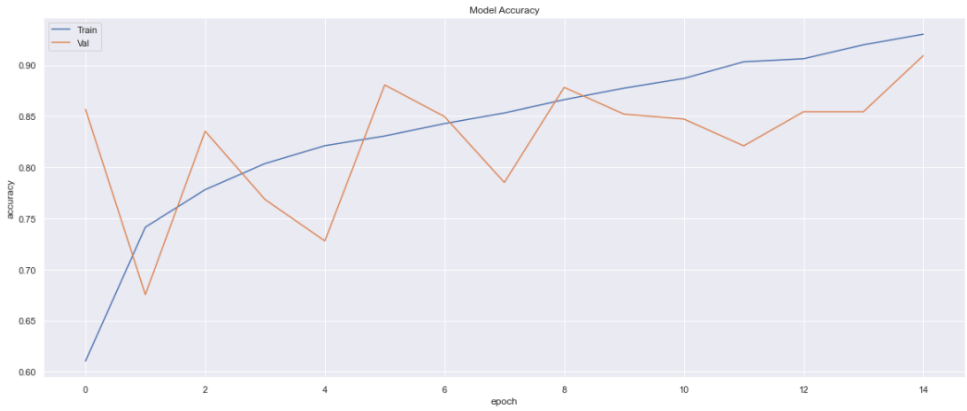

Figure 8: Training and validation accuracy over every epoch in eye diseases image

Enhanced Web Application Experience:

End users interacting with our web application cn effortlessly choose between chest X-ray and eye disease predictions, thanks to the shared CNN model. This streamlined approach not only simplifies user interaction but also optimizes computational resources for a more efficient and responsive application.

Chapter 4

RESULTS AND DISCUSSION

4.1 Results

4.1.1 Support Vector Machine

For Pneumonia: The results obtained from the HOG method show that the overall accuracy achieved is 76.44%. When considering different SVM kernels, it is observed that the poly kernel performs the best with a test accuracy of 78.53%, followed by the linear kernel with a test accuracy of 77.24%. The rbf and sigmoid kernels achieve test accuracies of 76.44% and 74.36% respectively. These results suggest that the choice of SVM kernel has an impact on the classification performance of the HOG method, with the poly and linear kernels yielding the highest accuracies.

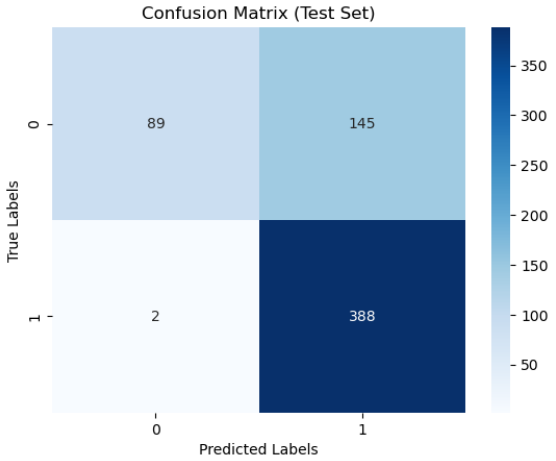

Figure 9: Confusion matrix for HOG method

On the other hand, the results obtained from the LBP method show an overall accuracy of 72.12%. Among the different SVM kernels, the poly kernel achieves the highest test accuracy of 73.24%, followed by the RBF kernel with a test accuracy of 72.12%. The linear and sigmoid kernels yield lower test accuracies of 62.50% and 61.70% respectively. These results indicate that the choice of SVM kernel also affects the classification performance of the LBP method, with the poly and RBF kernels performing better compared to the linear and sigmoid kernels.

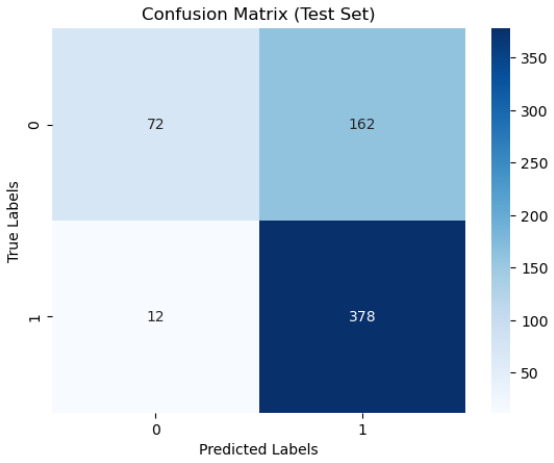

Figure 10: Confusion Matrix for LBP method

In conclusion, the SVM classifier achieved an accuracy of 76.44% with the HOG feature extraction method and an accuracy of 72.12% with the LBP feature extraction method. The poly kernel demonstrated the highest accuracy among the tested SVM kernels, with a test accuracy of 78.53% for HOG and 73.24% for LBP. The linear and RBF kernels also yielded reasonably good accuracies, while the sigmoid kernel showed lower performance. These results suggest that the combination of the SVM classifier and the HOG feature extraction method is more effective in accurately classifying the given dataset compared to the LBP method.

For eye-diseases: The results obtained from the HOG method show that the overall accuracy achieved is 78.64%. When considering different SVM kernels, it is observed that the linear kernel performs the best with a test accuracy of 82.94%, followed by the poly kernel with a test accuracy of 79.50%. The sigmoid kernel achieves a test accuracy of 76.87%, while the rbf kernel performs slightly lower with a test accuracy of 75.26%. These results suggest that the choice of SVM kernel has a significant impact on the classification performance of the HOG method, with the linear kernel yielding the highest accuracy.

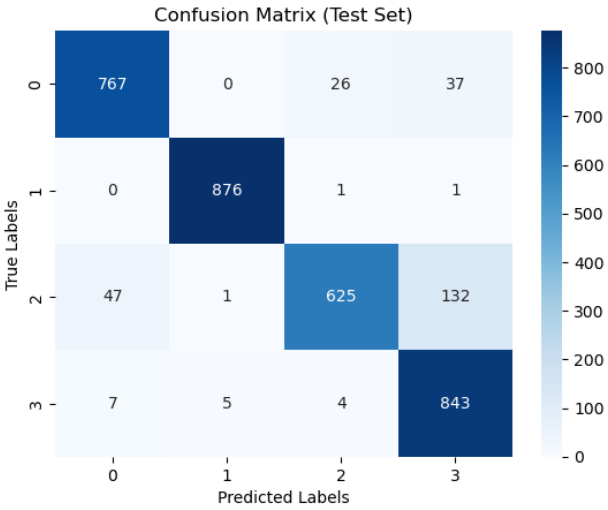

Figure 11: Confusion Matrix for HOG method

On the other hand, the results obtained from the LBP method show an overall accuracy of 77.64%. Among the different SVM kernels, the linear kernel achieves the highest test accuracy of 80.94%, followed closely by the poly kernel with a test accuracy of 78.50%. The sigmoid kernel performs slightly lower with a test accuracy of 76.87%, while the rbf kernel shows the lowest performance with a test accuracy of 74.26%. These results indicate that, similar to the HOG method, the choice of SVM kernel also affects the classification performance of the LBP method

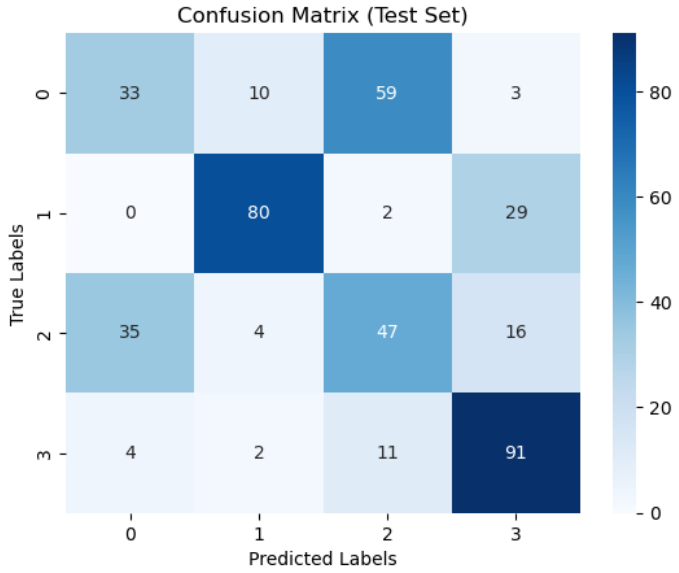

Figure 12: Confusion Matrix for LBP method

In conclusion, the SVM classifier achieved an overall accuracy of 78.64% with the HOG feature extraction method and an overall accuracy of 77.64% with the LBP feature extraction method. The linear kernel demonstrated the highest accuracy among the tested SVM kernels for both HOG and LBP methods. These results suggest that the combination of the SVM classifier and the HOG feature extraction method slightly outperforms the LBP method in accurately classifying the given eye disease dataset.

4.1.2 Naive Bayes

The success of our Naive Bayes model was varied based on our feature extraction technique. The discrete cosine transform was the most successful, with the 4 x 4 window transformation resulting in an overall accuracy of 83.65%.

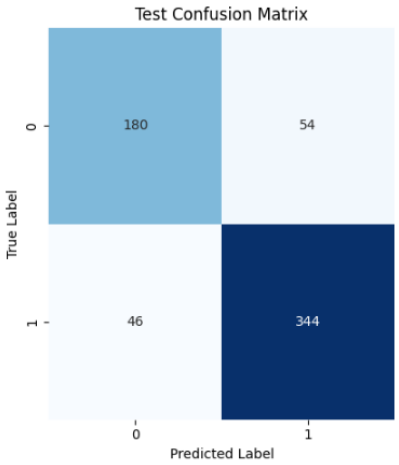

Figure 11: Naive Bayes Confusion Matrix

4.1.3 Transfer Learning

For Pneumonia: As our pretrained model is very large, when we were training it on our dataset it takes around four to five hours. Whatever our model consistently provided better performance, with an overall accuracy of 88.94% across our test data set. It also well performed to classifying normal and pneumonia. In this case, it was showing less overfitting signs, so we decided to use 20% dropout regularization to reduce overfitting issue. Then we got overall accuracy 87.50% across our test data set. Since we have used dropout regularization our model performed well generalized. The confusion matrix for both is shown below.

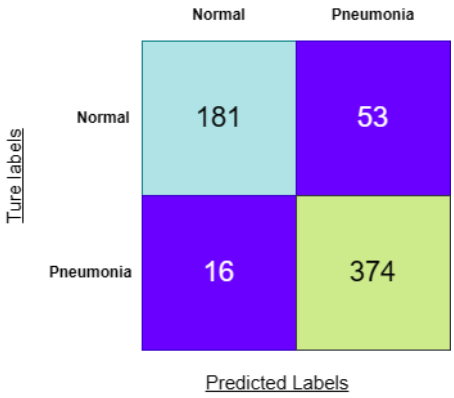

Figure 12: Confusion Matrix of Transfer Learning without dropout regularization

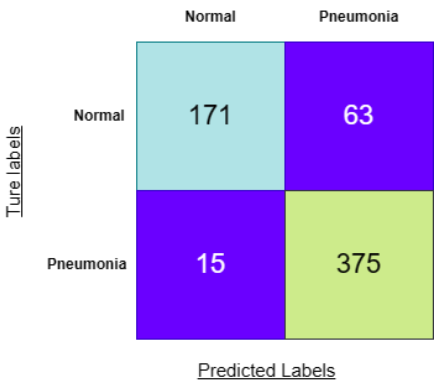

Figure 13: Confusion Matrix of Transfer Learning with dropout regularization

After using an oversampled dataset, we simply changed the data augmentation section, especially the random crop section. We increased the random crop property value from 180 to 190, and we ran our model again with the change, and we got 89.58% accuracy.

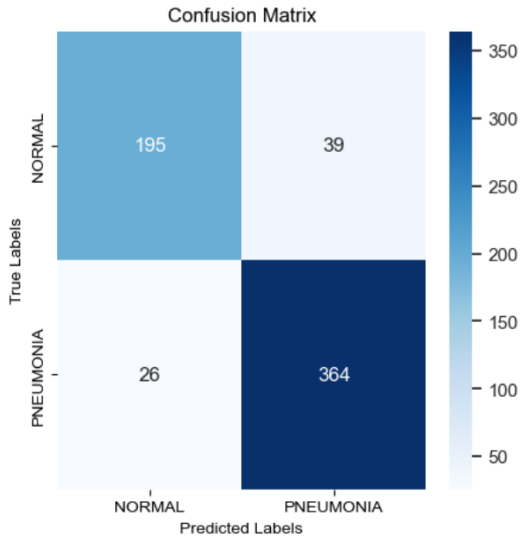

Figure 14: Transfer Learning Confusion Matrix

For Eye-disease: Our pretrained model is quite massive, therefore it takes about five hours to train it using our dataset. With an accuracy of 76.29% overall over our test data set, our model regularly performs on average. In order reduce the overfitting problem, we choose to employ 20% dropout regularization. Dropout regularization has allowed us to get well generalized model. Below is the confusion matrix.

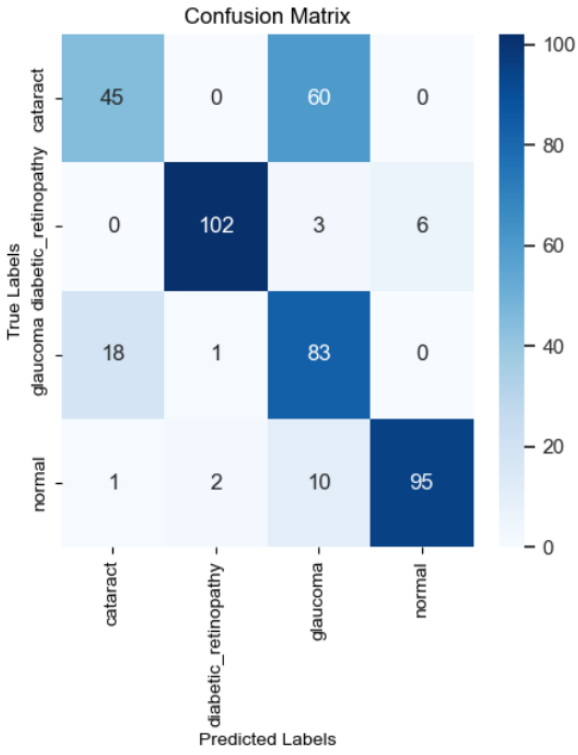

Figure 15: Eye Disease Confusion Matrix

4.1.4 Convolutional Neural Network

For Pneumonia: Our Convolutional Neural Network (CNN) model surpassed the performance of simpler models such as Naive Bayes and SVM, as anticipated.

Through the incorporation of data augmentation and MaxPooling layers, we effectively mitigated overfitting concerns. Notably, the CNN achieved a remarkable test accuracy of 78.6%, with a closely corresponding training accuracy of 92.1%. This demonstrated the CNN's superior capability in accurately classifying pneumonia cases compared to other models in our training set. Below is the confusion matrix.

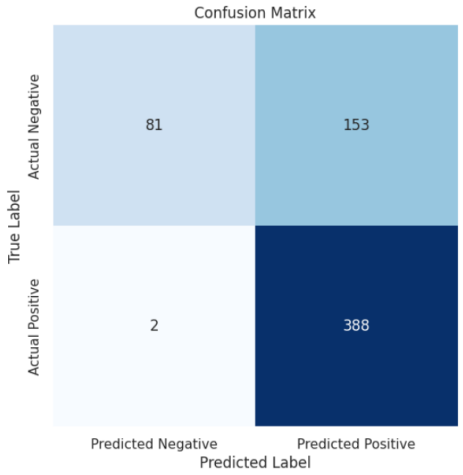

Figure 16: Pneumonia CNN Confusion Matrix

For Eye Disease: The CNN's ability to perform effectively in both chest X-ray and eye disease prediction underscores its versatility. With accuracy metrics of 68.02% for eye diseases respectively, the model contributes significantly to the overall predictive system. Below is the confusion matrix:

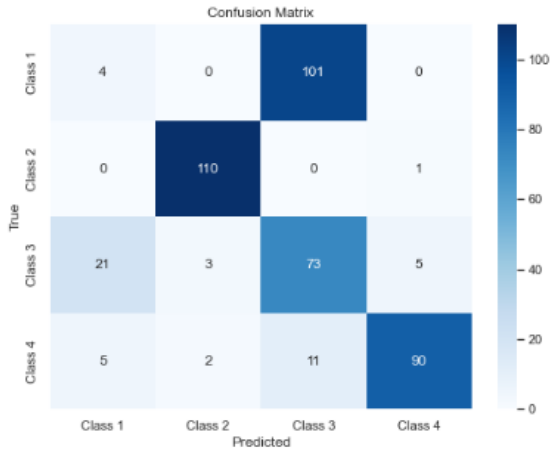

Figure 17: Eye Disease CNN Confusion Matrix

Chapter 5

WEB APPLICATION

We develop an innovative web application leveraging machine learning to predict health conditions from medical images. The primary focus was on chest X-ray images and eye disease images, with implementing models based on Naive Bayes and Convolutional Neural Networks (CNN), and transfer learning algorithm to enhance predictive accuracy and user interactivity.

5.1 Web Application Development

We translated our models into practical utility by constructing a user-friendly web application using Streamlit. This platform allows users to upload medical images and obtain predictions for various health conditions, including pneumonia and eye diseases.

5.2 User Interaction and Model Selection

The web application provides users with the flexibility to choose between Naive Bayes, CNN, or transfer learning models for predictions. Users can upload images and receive accurate predictions for pneumonia or various eye diseases, contributing to a user-centric healthcare solution.

5.3 Screen-shots

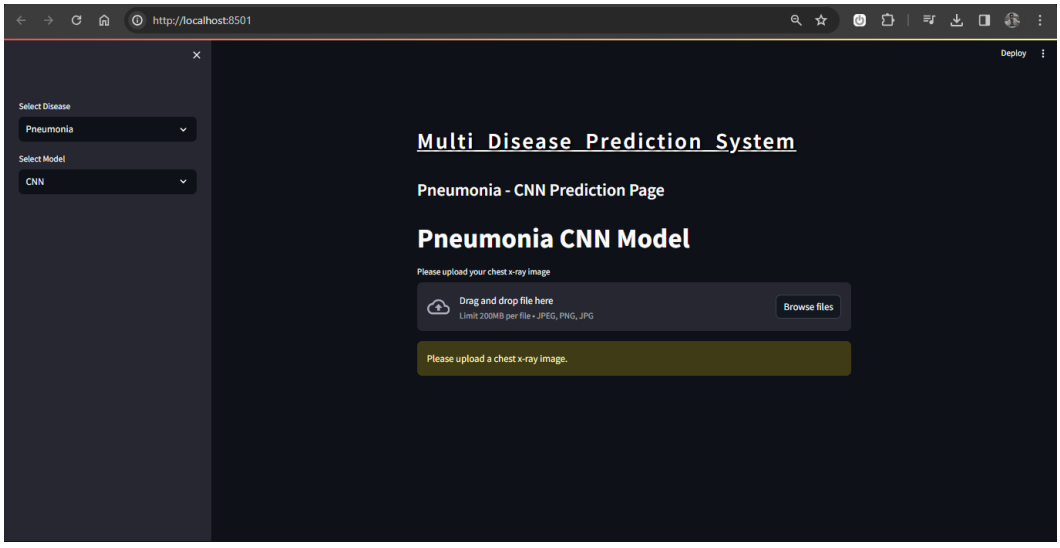

Figure 18: Home page

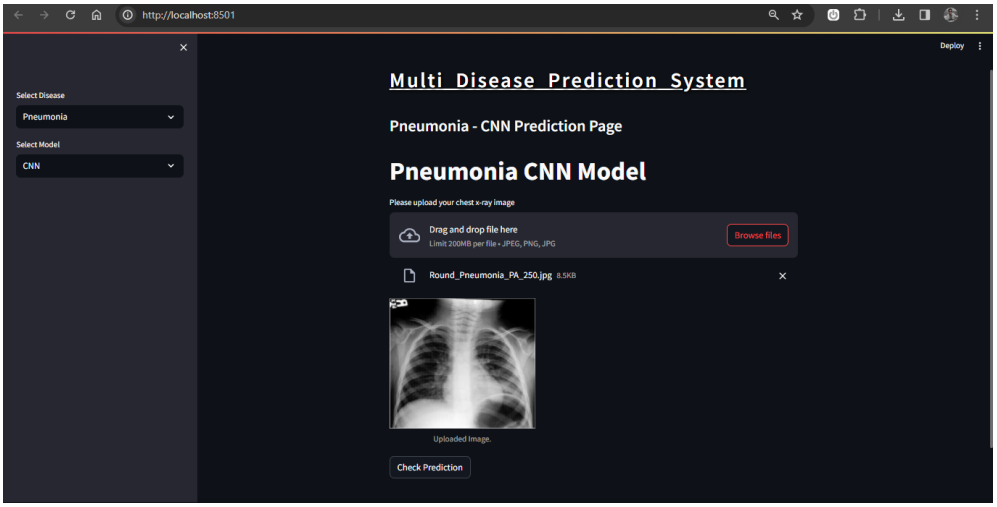

Figure 19: After uploading an image

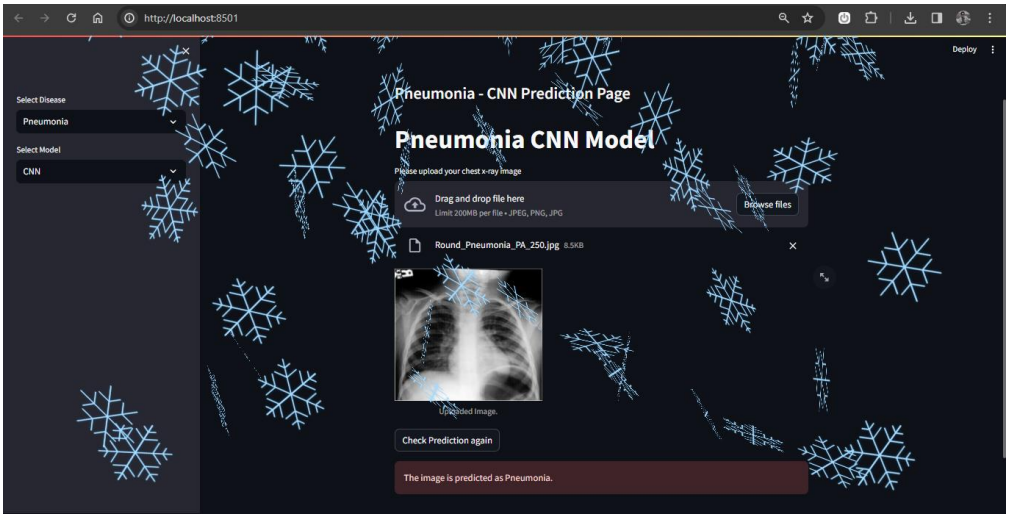

Figure 20: After checking the uploaded image

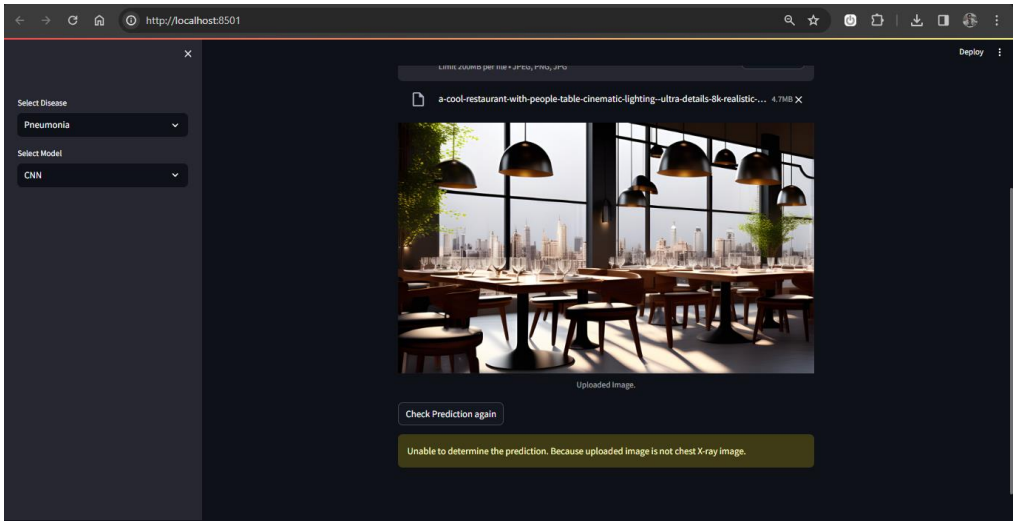

Figure 21: with random image our application will send a message that ‘Unable to determine the prediction. Because uploaded image is not chest X-ray image.’

Chapter 6

CONCLUSION

In conclusion, our experimental results demonstrate the effectiveness of different machine learning approaches for the classification of pneumonia and normal chest X-ray images. The SVM classifier with HOG feature extraction achieved an accuracy of 76.44%, outperforming the LBP feature extraction method with an accuracy of 72.12%. Among the SVM kernels, the poly kernel showed the highest accuracy for both HOG (78.53%) and LBP (73.24%). This indicates that the combination of SVM and HOG features is more suitable for accurately classifying the given dataset.

On the other hand, the Naive Bayes model with the discrete cosine transform achieved an impressive overall accuracy of 83.65%. The transfer learning approach using a pretrained model showcased even better performance, with an overall accuracy of 88.94% without dropout regularization and 87.50% with dropout regularization.

The transfer learning approach emerged as the highlight, exhibiting superior accuracy of 88.94% for pneumonia and consistent reliability in eye disease prediction. Its efficacy underscores the significance of leveraging pretrained models for medical image analysis.

The Convolutional Neural Network (CNN) not only surpassed simpler models but also showcased remarkable adaptability. With a test accuracy of 78.6% for pneumonia and 68.02% for eye diseases, the CNN proved to be a versatile solution for diverse healthcare applications.

The development of our user-friendly web application, featuring Naive Bayes, CNN, and transfer learning models, reflects our commitment to accessible and accurate health predictions. Users can effortlessly upload medical images, fostering a user-centric healthcare solution.

In conclusion, this project provides a solid foundation for future endeavors, encouraging further exploration of expanded datasets and advanced architectures to enhance predictive accuracy in the realm of healthcare solutions.

References

[1] Caselles V., Lisani L., Morel M., and Sapiro G., “Shape Preserving Local Contrast Enhancement,” *in Proceedings of International Conference Image Processing*, pp. 314 317, 1997.

[2] Chen, S.D., Ramli, A.R., “Contrast enhancement using recursive mean-separate histogram equalization for scalable brightness preservation”, *IEEE Trans. on Consumer Electronics*, vol. 49, no. 4, pp. 1301–1309, 2003.

[3] R.C. Gonzalez and R.E. Woods. *Digital Image Processing*. Prentice Hall, Upper Saddle River, New Jersey, EUA, 3nd edition, January 2008.

[4] Ain, Qurat-ul, et al. "Bayesian classification using DCT features for brain tumor detection." International Conference on Knowledge-Based and Intelligent Information and Engineering Systems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[5] Park, Dong-Chul. "Image Classification Using Naive 287 Bayes Classifier." International Journal of Computer 288 Science and Electronics Engineering (IJCSEE) 4, no. 289 3 (2016)

[6] Huang, G., Liu, Z., Van Der Maaten, L. and Weinberger, K.Q., 2017. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[7] Wang, K., Jiang, P., Meng, J., & Jiang, X. (2022). Attention-based DenseNet for pneumonia classification. IRBM 43: 479–485.

[8] NASSIH, B., AMINE, A., NGADI, M., & HMINA, N. (2019). “DCT and HOG Feature Sets Combined with BPNN for Efficient Face Classification.” Procedia Computer Science, 148, 116-125

[9] Babaqi, T., Jaradat, M., Yildirim, A.E., Al-Nimer, S.H. and Won, D., 2023. Eye Disease Classification Using Deep Learning Techniques. arXiv preprint arXiv:2307.10501.