Enhancing Computer-Aided Detection Systems for Chest X-ray Abnormalities using Deep Learning Techniques

Jyotirmay Mishra
Research Scholar (Computer Science)
School of Engineering and Technology
Shri Venkateshwara University, Gajraula, UP, INDIA
Email: mjyotirmay@gmail.com

Dr. Parveen Kumar
Research Guide (Computer Science)
School of Engineering and Technology
Shri Venkateshwara University, Gajraula, UP, INDIA
Email: pk223475@gmail.com

Received: 09-10-2024 Revised: 20-10-2024 Accepted: 29-10-2024
Published: 19-11-2024

Abstract: The focus here revolves around enhancing computer-aided detection (CAD) systems, particularly in diagnosing chest X-ray abnormalities using deep learning, notably Convolutional Neural Networks (CNNs). The shift from traditional computer vision methods, reliant on manual feature crafting, to deep learning has marked a substantial transformation in addressing computer vision challenges. A significant hurdle in this domain is the scarcity of data, especially for rare diseases, potentially leading deep neural networks to encounter diminishing gradients with increased network depth. To counter this, data augmentation techniques are employed, mathematically transforming original images to create additional samples from limited datasets. These transformations encompass various methods like flips (horizontal and vertical), rotation, shearing, zooming, filtering, and scaling.

Keywords*:* Computer-aided detection, chest X-ray abnormalities, deep learning, Convolutional Neural Networks (CNNs).

1. INTRODUCTION

The application of deep learning techniques in medical image analysis, particularly through Convolutional Neural Networks (CNNs), has significantly progressed the automation of diagnoses in modern healthcare[1]. It’s a notable shift from traditional computer vision methods to deep learning, enhancing the potential for diagnosing chest X-ray abnormalities and other medical imaging tasks. The transition from Artificial Neural Networks (ANNs) to deep learning in medical image classification has been fuelled by factors like accessible large medical datasets, enhanced computational power, and the development of advanced algorithms tailored for training deep neural network classifiers. Deep learning shows promise in surpassing human capabilities in certain visual and auditory recognition tasks, making it a fitting solution for healthcare and medical imaging. The focus of this paper centers on leveraging deep learning technology, particularly for lung nodule detection, addressing the challenge of efficiently and accurately analyzing complex medical imaging tests.

Deep learning's integration into radiology is poised to enhance diagnosis accuracy and support radiologists in providing precise interpretations. However, the goal isn't to replace radiologists but to augment their capabilities and streamline repetitive tasks. While deep learning has made significant strides in medical imaging, several challenges persist. Data augmentation, particularly in generating augmented images from original annotated images, remains an area ripe for further research and development, given the current trends [2,3].

Advancements in the intersection of hardware, software, and deep learning techniques are reshaping radiological image prediction and evaluation, particularly in lesion detection and assessment. However, several persistent challenges persist in this domain. Firstly, deep learning models are prone to overfitting due to their sensitivity to both the quality and quantity of training data. Robust evaluation across diverse disease occurrences and imaging technologies is imperative. Secondly, the inherent black box nature of deep learning systems presents hurdles in understanding their underlying mechanisms, raising concerns about detecting failures in unusual disease conditions. Moreover, ethical and legal considerations surrounding the use of medical image data in commercial deep learning systems demand attention. The reliance of these systems on training data quality emphasizes this concern. Furthermore, the design and architecture of deep learning models can lead to overfitting with limited dataset sizes, emphasizing the need for appropriate model architectures [3, 4, 5, 6]. Additionally, limitations exist in relying solely on frontal radiographs for diagnosis and constraints in utilizing patient symptom history, potentially limiting radiographs' accuracy in certain scenarios[5]. Addressing these issues stands as a pivotal step for ongoing advancements in deep learning for medical imaging.

2. CHEST X- RAY CHARACTERISTICS AND ABNORMAL

Normal chest X-ray images typically display distinct bony structures, notably the ribs, depicted as white structures against a dark backdrop. The symmetry of the lung region, including the shoulders and arms, is a prominent feature. Abnormalities in chest X-rays often manifest as changes in texture, size, and shape within the lung area[21]. Analyzing multi-scale shape features, edges, and textures assists in identifying these irregularities.Chest X-ray examinations, also known as CXR, chest radiographs, or chest roentgenograms, offer a straightforward, cost-effective, and relatively safe means of imaging the chest and its internal organs. Minimal radiation exposure makes it a preferred diagnostic tool. The shadows captured on film result from radiation passing through the chest cavity and interacting with the density of the organs. The darkness or intensity of these shadows correlates with the density of the internal structures.

These images serve as valuable diagnostic tools for various conditions, including pneumonia, lung masses or nodules, rib fractures, effusion, tuberculosis, cardiomegaly, pneumothorax, and congestive heart failure. Analyzing these images aids in detecting and diagnosing a wide range of abnormal chest conditions.

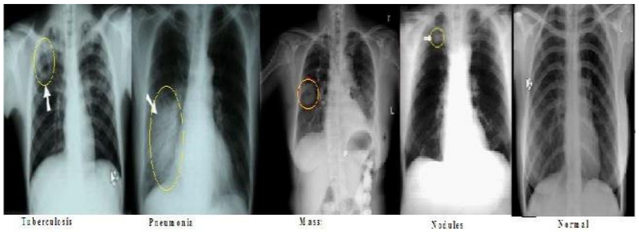


Figure 1: Chest X-ray images with Tuberculosis, Pneumonia, Mass, Nodules and Normal

Tuberculosis: In chest X-rays, tuberculosis often presents as lung cavities resembling scratches within the lung tissue. The findings associated with active pulmonary tuberculosis reveal infiltrated airspaces within the lung parenchyma. These infiltrations typically appear dense or patchy with irregular, hazy borders (figure 1).

Nodules: Identified as rounded edges in chest X-ray images, nodules often manifest as irregularly bordered spots. Single pulmonary nodules seen on chest X-rays generally exhibit an irregular margin surrounded by small settlements, typically with a diameter ranging between 8mm to 10mm. Detecting nodules smaller than 8mm in diameter can be challenging. Larger nodules, especially those with irregular shapes situated in the upper lung regions, may signify the possibility of malignancy.

Mass: Lung masses display irregular and distinct shadows on the lung fields, presenting sharp-edged lesions that appear homogeneous. These masses often appear as lesions with homogenous and clear margins, differing from the appearance of diffuse infiltrates.

Pneumonia: Chest X-rays depicting pneumonia exhibit abnormal misty white shadows in the lung fields, contrasting with the darker areas. In a vertical film, this could cause an effusion that appears as a blunting on the lateral costophrenic, expanding enough to become visible on the posterior costophrenic sulcus. Detecting an effusion in a PA film typically requires around 200 ml of fluid, while approximately 75 ml of fluid in the lateral view of the AP film can indicate the presence of an effusion. A lateral decubitus film proves useful in confirming the presence of a fluid collection on the dependent side, often presenting as a graded haze with a denser base.

3. CLASSIFICATION METHODOLOGY

The CNN methodology is described below with the first step as data preparation.

A. Data Preparation

There are popular datasets. These are explained below:

1) Chest X-Ray8

(Xiaosong Wang, et al (2017)) introduced the ChestX-ray8 database [15], The dataset comprises three main folders—train, test, and val—each containing subfolders categorizing X-Ray images into two groups: Pneumonia and Normal. There's a total of 5,863 JPEG images across these categories, focusing on chest X-rays taken in an anterior-posterior position. These images were sourced from pediatric patients aged one to five years old at the Guangzhou Women and Children’s Medical Center in Guangzhou. The collection process involved routine clinical care where chest X-ray imaging was conducted.

2) NIH Chest X-rays

The National Institutes of Health (NIH) Chest X-ray Dataset stands as a pivotal resource in medical imaging, encompassing 112,120 chest X-ray images annotated with disease labels from 30,805 distinct patients [16]. Remarkably, this dataset addresses a critical gap in the medical field by leveraging Natural Language Processing (NLP) techniques to extract disease classifications from associated radiological reports, enabling the creation of labels that boast >90% accuracy, deemed suitable for weakly-supervised learning endeavors. Despite limitations such as potential erroneous labels due to NLP extraction and scarcity of disease region bounding boxes, this extensive dataset, provided in 12 ZIP files varying in size from ~2 GB to 4 GB, facilitates groundbreaking research in computer-aided detection and diagnosis (CAD) for thoracic diseases. With 15 classes encompassing various pathologies, including "No findings" and a spectrum of diseases like pneumonia, pneumothorax, and cardiomegaly, this dataset not only empowers researchers but also encourages collaborative contributions to enhance its accuracy and utility. Furthermore, a smaller subset comprising 5% of the images is made available for ease of use in kernels, fostering innovation and analysis within the Kaggle platform.

B. Pre-Processing

The input images undergo preprocessing to improve contrast and eliminate extraneous areas. Lung X-ray data can be subject to various sources of variation that might impact classification performance, including differences in pixel size and contrast. Initially, all images were resized to 512 x 512 pixels and subsequently converted to grayscale.

Understanding the distribution of diagnoses within the dataset is crucial for model training and evaluation as shown in figure 2. The bar plot showcasing the prevalence of pneumonia samples, constituting approximately 73% of the dataset, reveals a significant imbalance in class distribution. This imbalance, with pneumonia samples dominating the dataset, could potentially simplify the model's learning process in identifying pneumonia cases. The substantial representation of pneumonia instances suggests that the model might inherently learn pneumonia-related patterns without the need for specific class weight adjustments.

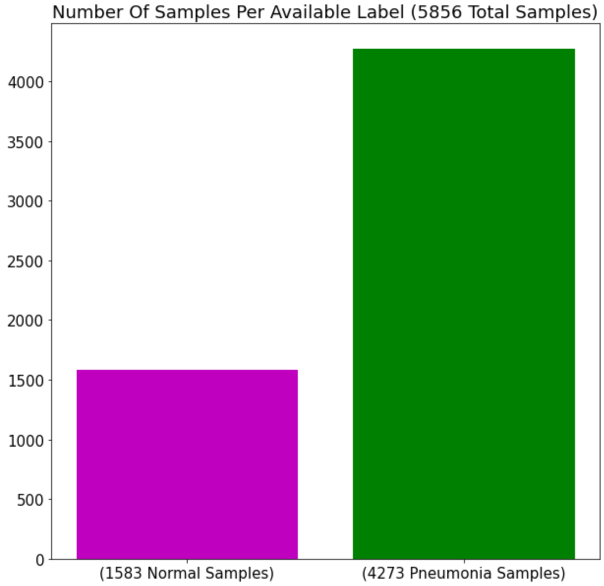
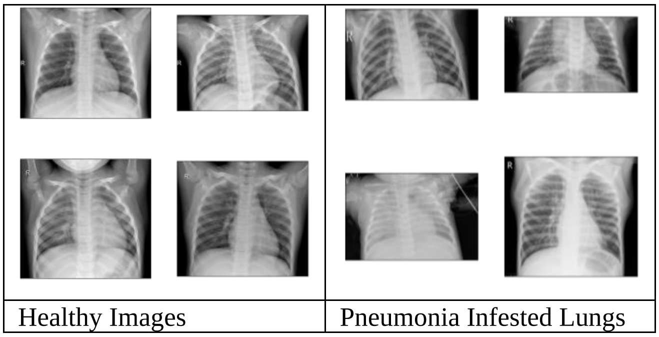


Figure 2: Label distribution in the dataset.

Employing a 70% training set alongside a 15% validation set and 15% test set proportion facilitates a comprehensive evaluation setup. This division allows ample data for training while dedicating significant portions to validation and testing, ensuring the model's performance is rigorously assessed across various datasets. This methodical approach enhances the model's reliability and its ability to generalize to unseen data, vital for accurate pneumonia identification in chest X-ray images (Figure 3).



Healthy Images

Pneumonia Infested Lungs

Figure 3: Healthy Vs Unhealthy Images in Dataset

C. Data Augmentation

We utilize various augmentation strategies to generate new training sets. Each new training set is simply the original training images in addition to the training images augmented by one of the techniques below. All the original and augmented images in the training and validation set are cropped to 512 X 512 pixels. Then, the images in training sets are trained using CNNs.

D. Model Implementation

The augmented image data sets are executed by the Convolutional Neural Network, it consists of three convolutional layers where batch size 64 and buffer size set 128. Following the convolutional layers are the three pooling layers performing max pool function having stride value as

Finally these layers are followed by two fully connected layers. The last fully connected layer FCL2 consists of two nodes equal to the size of classes. Evaluations metrics are calculated by computing classification accuracy. The created model trained between 15 iterations, the maximum accuracy obtained is considered as a result. The performances are evaluated on the validation set. Therefore, the Convolutional network predicts any of the two classes for the images.

Setting up ImageDataGenerator objects for batch input, incorporating data augmentation specifically for the training data, is a robust strategy to tackle overfitting while training the model. By performing data augmentation solely on the training set, you're enhancing its diversity without altering the validation and test sets, ensuring they remain untouched and unbiased for accurate evaluation.

The parameters selected for data augmentation are designed to create realistic transformations, mimicking the variations and scenarios the model might encounter in real-world usage. The objective is to present the model with diverse yet plausible images during training, thereby improving its generalization capabilities. Creating DataFrameIterator objects via the flow_from_dataframe method further streamlines the process. By specifying parameters like target image size (512 x 512), grayscale color mode, batch size (64), class mode (binary), and shuffling preferences (True for training and False for validation & test), we are defining a structured way for the iterator to access, preprocess, and provide the images and their corresponding binary labels during model training.

This setup ensures that during training, the model receives batches of 64 preprocessed images resized to the specified dimensions along with their binary labels, effectively preparing the data for the model's learning process. Meanwhile, for validation and test data, the data remains in its original form, scaled to a range between 0 and 1, ensuring consistency and reliability during model evaluation without introducing variations through augmentation. Activation functions are crucial for neural networks, impacting performance and training dynamics. Linear units provide a range of activations but lack dependence on input for backpropagation. ReLU has gained popularity for its success, particularly in CNNs. This study explores various activation functions in a CNN classifier, aiming to validate their effectiveness in classification accuracy and average loss for medical images. Separate filters for each input channel occur in the second convolutional layer, generating a 4D output tensor. Fully-connected layers reduce this to 2D. Cross-entropy, a performance measure in classification, aims to minimize error by adjusting network variables. TensorFlow provides functions for computing cross-entropy to optimize model accuracy.

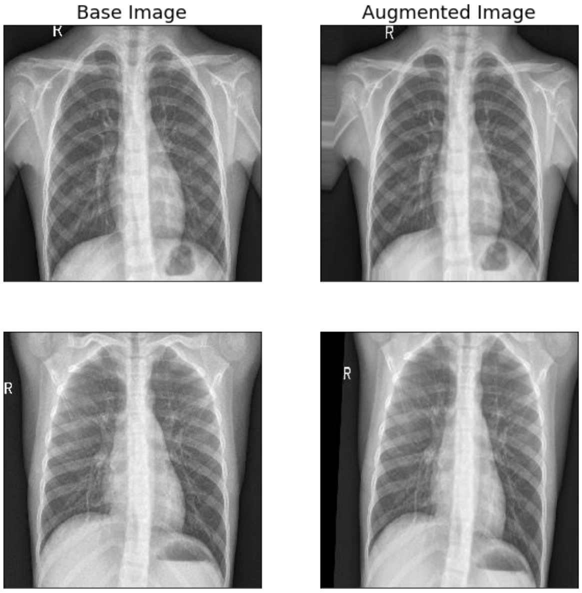


Figure 4: Base Images Vs. Augmented Images

E. Evaluation Measures

The experiments involved two types of estimators: pretrained CNN Classifier (Premade estimators) and a proposed CNN classifier (customized estimator) (Figure 4). Both sets of parameters were fine-tuned for each execution, evaluating their respective performances using various evaluation measures. Following are the key concepts in machine learning evaluation:

- Accuracy and loss are fundamental metrics in evaluating machine learning algorithms.
- Accuracy measures the discrete true or false values for the training data's categories, while loss (like cross-entropy) is used for optimizing the algorithm.
- While accuracy is useful for computing categorical training data metrics, it's not differentiable and isn't suitable for backpropagation. Loss functions are better for backpropagation as they're differentiable.
- The primary goal of the learning model is to reduce the loss value concerning the model parameters by altering weight vectors through optimization techniques like backpropagation.
- Evaluation involves testing the model with unseen data, recording mistakes (zero-one loss), and computing the percentage of misclassifications.
- Overfitting occurs when the model memorizes the training data and fails with new, unseen data. Regularization helps prevent this by limiting the model's complexity.

In multilabel classification, comparing the predicted labels to the true labels determines subset accuracy, with a perfect match scoring 1.0 and any mismatch resulting in a score of 0.0. The study aims to explore various deep learning architectures, particularly CNNs, for improving medical image classification by leveraging their ability to learn relevant features and reduce the number of required parameters. The study seeks to serve as a resource for learners and researchers interested in this field, focusing on tuning neurons to enhance the classifier model's performance in medical image classification.

F. CNN classification of medical images

This architecture represents a Convolutional Neural Network (CNN) designed for a specific classification task. Here's a breakdown of the layers and their functions:

- Input Layer (Input): Accepts input images with a shape of (512, 512, 1), indicating images of size 512x512 with a single channel (grayscale).
- Convolutional Layers (Conv2D): A sequence of convolutional layers (with 2D convolution operation) that apply 2D filters to the input data to create feature maps. These layers learn various features from the input images. Each convolutional layer has different numbers of filters (64, 96, 128, 160, 192, 224, 256) and different output shapes as shown in the architecture.

- MaxPooling Layers (MaxPooling2D): Following each convolutional layer, max-pooling layers are applied to down-sample the feature maps by taking the maximum value in each window. This reduces the spatial dimensions (width and height) of the data.
- Dropout Layers (Dropout): Introduces dropout, a regularization technique where a fraction of input units are randomly set to zero during training. This helps prevent overfitting by reducing the interdependent learning between neurons.
- Global Average Pooling Layer (GlobalAveragePooling2D): Reduces the spatial dimensions of the feature maps to a single vector by taking the average of each feature map. This prepares the data for the final classification layer.
- Output Layer (Dense): The final layer is a densely connected (fully connected) layer with a single neuron, representing the output. It uses a sigmoid activation function (commonly used in binary classification tasks) to produce a single output value between 0 and 1.
- Total Parameters: Indicates the total number of trainable parameters in the network, which includes weights and biases.
- Trainable Parameters: Represents the parameters that can be updated during training to minimize the loss.
- Non-trainable Parameters: Denotes parameters that are not updated during training, often associated with operations like pooling or activations.

This architecture comprises multiple convolutional layers with max-pooling, dropout for regularization, and ends with a global average pooling layer before the final output layer for binary classification. The goal is to learn hierarchical features from the input images and make predictions based on those learned features.

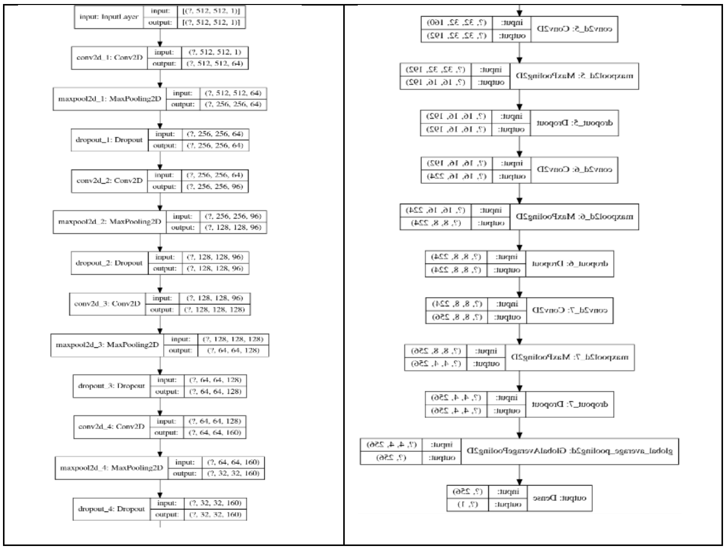

input: InputLayer | input: [(?, 512, 512, 1)] | output: [(?, 512, 512, 1)]
conv2d_1: Conv2D | input: (?, 512, 512, 1) | output: (?, 512, 512, 64)
maxpool2d_1: MaxPooling2D | input: (?, 512, 512, 64) | output: (?, 256, 256, 64)
dropout_1: Dropout | input: (?, 256, 256, 64) | output: (?, 256, 256, 64)
conv2d_2: Conv2D | input: (?, 256, 256, 64) | output: (?, 256, 256, 96)
maxpool2d_2: MaxPooling2D | input: (?, 256, 256, 96) | output: (?, 128, 128, 96)
dropout_2: Dropout | input: (?, 128, 128, 96) | output: (?, 128, 128, 96)
conv2d_3: Conv2D | input: (?, 128, 128, 96) | output: (?, 128, 128, 128)
maxpool2d_3: MaxPooling2D | input: (?, 128, 128, 128) | output: (?, 64, 64, 128)
dropout_3: Dropout | input: (?, 64, 64, 128) | output: (?, 64, 64, 128)
conv2d_4: Conv2D | input: (?, 64, 64, 128) | output: (?, 64, 64, 160)
maxpool2d_4: MaxPooling2D | input: (?, 64, 64, 160) | output: (?, 32, 32, 160)
dropout_4: Dropout | input: (?, 32, 32, 160) | output: (?, 32, 32, 160)
conv2d_5: Conv2D | input: (?, 32, 32, 160) | output: (?, 32, 32, 192)
maxpool2d_5: MaxPooling2D | input: (?, 32, 32, 192) | output: (?, 16, 16, 192)
dropout_5: Dropout | input: (?, 16, 16, 192) | output: (?, 16, 16, 192)
conv2d_6: Conv2D | input: (?, 16, 16, 192) | output: (?, 16, 16, 224)
maxpool2d_6: MaxPooling2D | input: (?, 16, 16, 224) | output: (?, 8, 8, 224)
dropout_6: Dropout | input: (?, 8, 8, 224) | output: (?, 8, 8, 224)
conv2d_7: Conv2D | input: (?, 8, 8, 224) | output: (?, 8, 8, 256)
maxpool2d_7: MaxPooling2D | input: (?, 8, 8, 256) | output: (?, 4, 4, 256)
dropout_7: Dropout | input: (?, 4, 4, 256) | output: (?, 4, 4, 256)
global_average_pooling2d: GlobalAveragePooling2D | input: (?, 4, 4, 256) | output: (?, 256)
output: Dense | input: (?, 256) | output: (?, 1)

Figure 5: The Proposed CNN Architecture

G. Implementation:

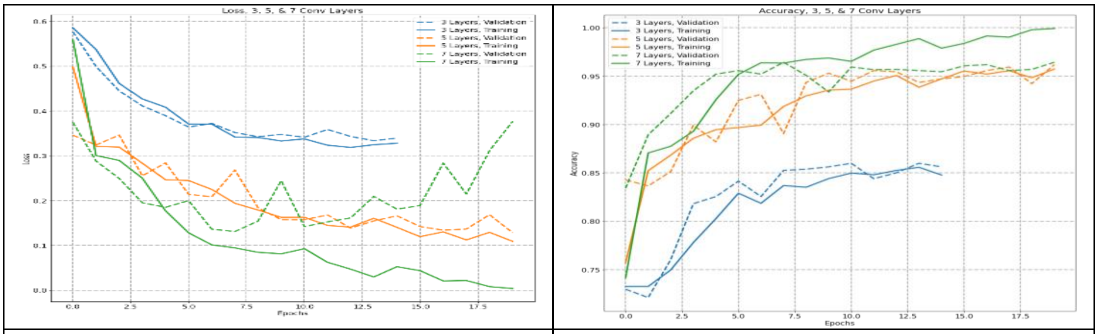
Experiments are performed over nine different augmented image set; the various augmented images are stored in different folders which are labeled by their corresponding augment type. Every augmented dataset consist of augmented images of various pathology and the normal images. The augmented images are again preprocessed for resizing it to 512X512 pixels. The experiments are executed repeatedly with various augmented image dataset for obtain any of the five classes. The various augmentation techniques applied are described below (Figure 5).

The augmented image data sets are executed by the Convolutional Neural Network, it consists of three convolutional layers where batch size 64 and buffer size set 128. Following the convolutional layers are the three pooling layers performing max pool function having stride value as 2. Finally these layers are followed by two fully connected layers. The last fully connected layer FCL2 consists of two nodes equal to the size of classes. Evaluations metrics are calculated by computing classification accuracy. The created model trained between 15 iterations; the maximum accuracy obtained is considered as a result. The performances are evaluated on the validation set. Therefore, the Convolutional network predicts any of the two classes for the images.

Using the ImageDataGenerator object for model input is a smart move, especially when dealing with image paths instead of image arrays. The flow_from_dataframe method is an ideal solution in this scenario, enabling the creation of DataFrames that contain image paths labeled under the "filename" column and corresponding diagnosis labels under the "class" column. This approach aligns perfectly with the requirements of the flow_from_dataframe method, allowing seamless feeding of the data into the model during training and validation stages. By structuring the data in this way, it will facilitate the model's access to the images and their respective labels, ensuring smooth and efficient processing while training the model to identify normal and pneumonia cases based on chest X-ray images.

4. RESULTS AND DISCUSSION

We have experimented on the proposed CNN using different configurations and schemes such as the optimum number of layers, results on unaugmented and augmented data, effect of regularization, effect of dropout, effect of variable Learning Rate. These outcomes are explained in detail in subsections.



Loss after 3, 5 and 7 layers

Accuracy after 3, 5 and 7 layers

Figure 6: Classifiers outcome after 3rd, 5th and 7th layer of the proposed CNN network

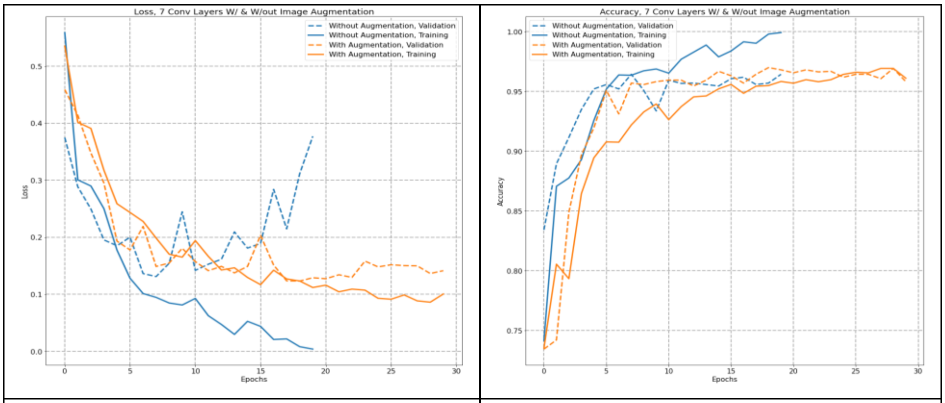
H. Layer wise outcome of the proposed CNN network

In Figure 6, we examine the output from different layers (3rd, 5th, and 7th) within the proposed CNN network. In 6(a), the graph illustrates both training and validation loss, while in 6(b), the accuracy achieved per epoch is displayed. Analyzing the loss values reveals that utilizing only 3 layers proves inadequate. With 5 layers, the network reaches a point of sufficiency but tends to plateau early, exhibiting higher loss values and lower accuracy. The 7-layer configuration performs well, yet beyond 10 epochs, signs of overfitting become apparent. Optimal performance is noted around 13 or 14 epochs, showcasing the model's good accuracy, approximately at 95.5%.

The provided code initializes an `ImageDataGenerator` object using Keras' `ImageDataGenerator` class. This generator is commonly used for data augmentation in image datasets during the training of neural networks. Let me explain the parameters used:

- **rescale:** This parameter normalizes the pixel values of the images to a range between 0 and 1 by dividing each pixel value by 255. It's a standard practice to scale pixel values to assist the model in learning better.
- **zoom_range:** It determines the random range for zooming in on the images. Here, a value of 0.1 signifies that the images can be zoomed in by a maximum of 10%.
- **height_shift_range:** It defines the range for randomly shifting the height of the images. A value of 0.05 indicates that the images can be shifted vertically by a maximum of 5%.
- **width_shift_range:** Similar to `height_shift_range`, this parameter sets the range for randomly shifting the width of the images by a maximum of 5%.
- **rotation_range:** It specifies the range within which the images can be randomly rotated. A value of 5 means that the images can be rotated to a maximum of 5 degrees clockwise or counterclockwise.

I. *Effect of image augmentation on training*



Loss after image augmentation | Accuracy after image augmentation

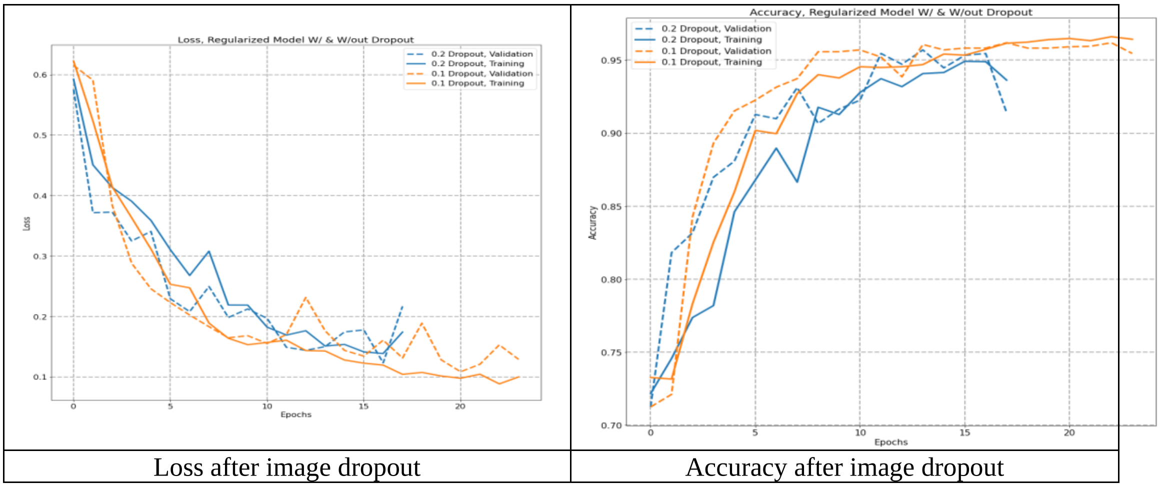
Figure 7: Classifiers outcome of the proposed CNN network after image augmentation

In Figure 7, the output from the CNN network following image augmentation is examined. In 7(a), the graph displays both training and validation loss, while in 7(b), the accuracy achieved per epoch is depicted. Comparison of the loss values after image augmentation indicates a slower decrease in training loss compared to unaugmented data. However, notably, the validation loss is lower for augmented images compared to the unaugmented dataset. The implementation of image augmentation significantly mitigates the overfitting issue. Validation accuracy aligns closely with training accuracy and experiences a nearly 1% improvement, reaching approximately 96.5%.

J. *Effect of Dropout Layer*

A Dropout layer is a regularization technique commonly used in neural networks, including convolutional neural networks (CNNs), to prevent overfitting and improve the network's generalization ability. It works by randomly deactivating or "dropping out" a fraction of neurons during training.

Overall, Dropout layers play a crucial role in regularizing neural networks, including CNNs, by preventing overfitting, promoting better generalization, and improving the network's ability to learn more robust and meaningful features from the data.



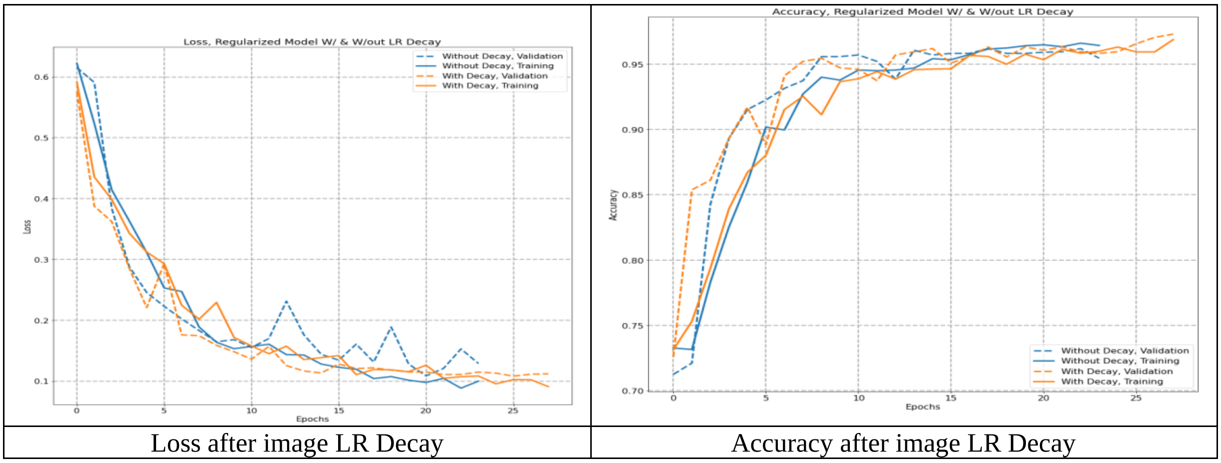
Loss after image dropout | Accuracy after image dropout

Figure 8: Classifiers outcome of the proposed CNN network after dropout layer

Figure 8 showcases the impact of incorporating a dropout layer within the CNN network. In 8(a), the graph exhibits both training and validation loss, while 8(b) displays the accuracy achieved per epoch. By analyzing the loss values, a slower decline in training loss is observed beyond 15 epochs without the dropout layer, indicating a potential risk of overfitting in the system. However, with the inclusion of the dropout layer, the loss continues to decrease without this concern, emphasizing its role in preventing overfitting. Particularly noteworthy is the consistently lower validation loss observed with the dropout layer. The utilization of the dropout layer effectively mitigates the overfitting problem, aligning validation accuracy closely with training accuracy and showing a notable improvement of nearly 0.5%, reaching an approximate accuracy of 96%.

K. Effect of learning Rate Decay

Learning rate decay is a technique used in training neural networks where the learning rate, which determines the step size taken during the optimization process (like gradient descent), is systematically reduced over time or after a certain number of training steps/epochs.



Loss after image LR Decay

Accuracy after image LR Decay

Figure 9: Classifiers outcome of the proposed CNN network with variation in learning rate

Overall, learning rate decay is a crucial technique in optimizing neural network training, ensuring a more stable convergence towards an optimal solution while preventing issues like overshooting and overfitting. The choice of decay schedule often depends on the problem domain, network architecture, and characteristics of the dataset.

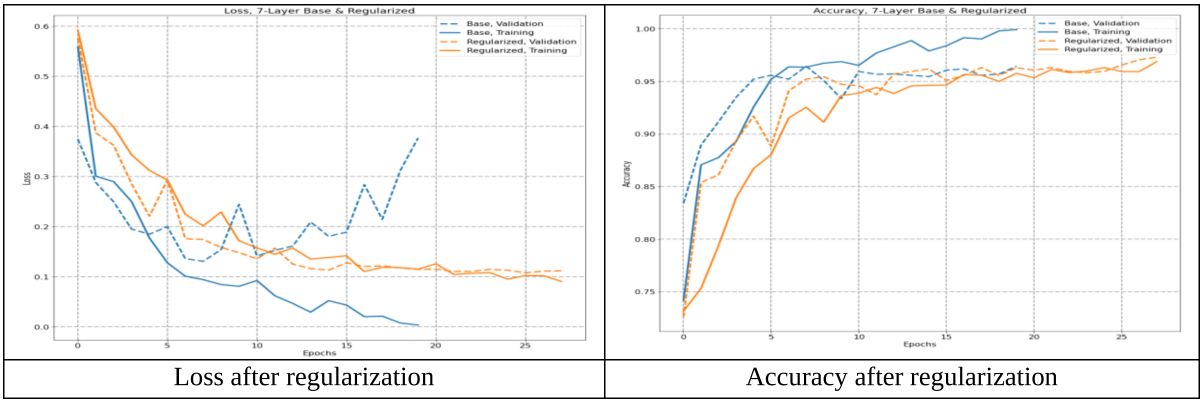
In Figure 9, the implementation of a variable learning rate within the CNN network is explored. In 9(a), the graph visualizes both training and validation loss, while 9(b) illustrates the accuracy attained per epoch. Upon scrutinizing the loss values, a consistent decrease in both training and validation loss is evident, contrasting with a fixed learning rate where substantial fluctuations in loss are observed. This suggests a smoother learning trajectory within the system when utilizing a constant learning rate. However, with the variable rate approach, there is not a clear indication of achieving high accuracy levels. The variable learning rate, while offering smoother learning patterns, seems to lack evidence of significant accuracy improvements compared to a fixed learning rate.

L. Effect of Regularization on Performance

In Figure 10, the impact of employing various regularization techniques such as image augmentation, dropout, and learning rate decay within the CNN is demonstrated. In 10(a), the graph portrays both training and validation loss, while 10(b) showcases the accuracy achieved per epoch. Comparing the regularized CNN to the non-regularized or Base CNN, notable improvements are observed.

The regularized CNN exhibits superior performance, showcasing a substantial validation accuracy of maximum 97% up to 20 epochs. Remarkably, the regularized model showcases a lack of overfitting, maintaining a consistent alignment between training and validation accuracy.

Given the absence of overfitting, there's an expectation that the accuracy may continue to rise with more epochs. The regularized CNN's robust performance suggests its effectiveness in preventing overfitting and enhancing the model's ability to generalize well to unseen data. This promising outcome signifies the success of incorporating various regularization techniques to improve the CNN's performance and achieve higher accuracy levels.



Loss after regularization

Accuracy after regularization

Figure 10: Classifiers outcome of the proposed CNN network with regularization

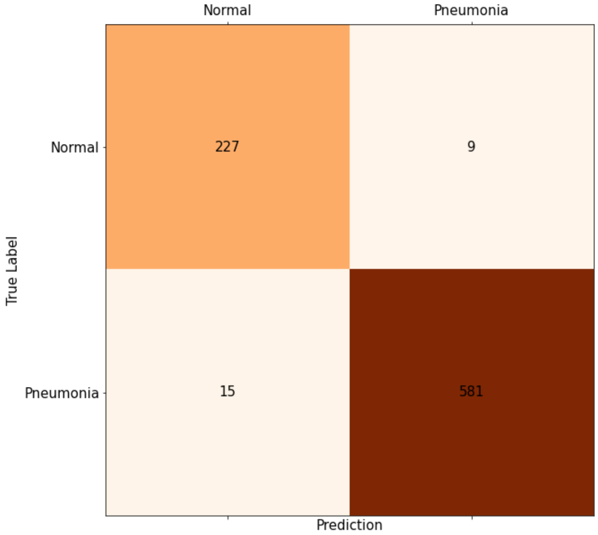


Figure 11: Confusion matrix for regularized CNN classifiers

Table 1: Outcome of the regularized classifier

	Precision	**Recall**	**F1-Score**
Class 0	0.94	0.96	0.95
Class 1	0.99	0.98	0.98

The precision, recall, and F1-score metrics offer a comprehensive assessment of a classification model's performance in distinguishing between two classes, Class 0 and Class 1 is shown in table I which has been extracted from confusion matrix of figure 11. A precision of 94% for Class 0 indicates that among all instances predicted as Class 0, 94% were accurately classified, highlighting a high level of correctness in these predictions. Concurrently, a precision of 99% for Class 1 signifies an even greater accuracy, with nearly all instances predicted as Class 1 being correct. Moving to recall, the model demonstrated a capability to capture 96% of all actual Class 0 instances and 98% of all Class 1 instances, indicating a strong ability to detect the majority of instances belonging to both classes. The F1-scores, harmonizing precision and recall, validate the model's overall performance, with Class 0 achieving a robust score of 0.95 and Class 1 displaying even higher performance with a score of 0.98. These metrics collectively affirm the model's accuracy, sensitivity, and balanced performance in correctly classifying instances from both classes, albeit with a slightly superior performance in Class 1 identification.

M. Outcome in Images

Adjusting the decision threshold within the [0, 1] range can significantly impact the model's prediction confidence and subsequently affect its performance across various evaluation metrics. Changing the threshold alters the balance between precision and recall, thereby influencing metrics like accuracy, precision, recall, and F1 score. For instance, a higher threshold might increase precision by reducing false positives but could lower recall by missing some true positives. Conversely, a lower threshold might elevate recall while potentially increasing false positives, consequently impacting precision.

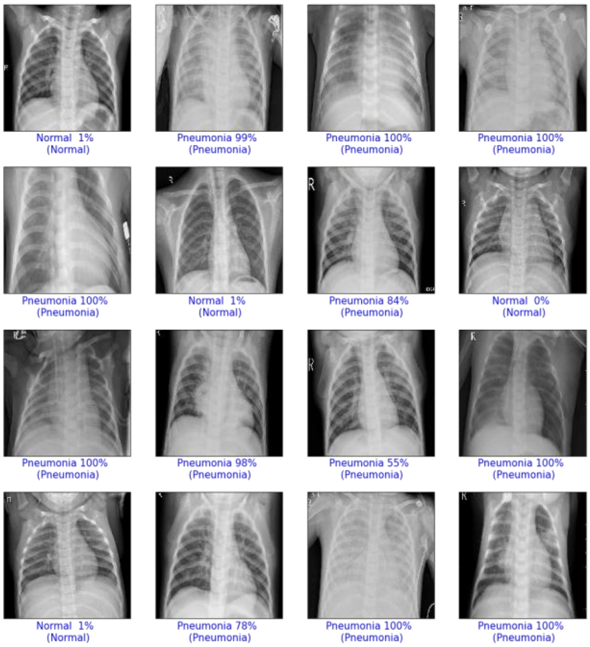


Figure 12: The CNN classifier output on images.

By systematically varying the threshold and observing how these metrics fluctuate, you can gain insights into the model's performance at different confidence levels. This analysis enables you to make informed decisions about the threshold selection, balancing the trade-offs between precision and recall based on the specific requirements or priorities of the classification task (figure 12).

Accuracy, being the overall measure of correct predictions over all instances, tends to be influenced by both precision and recall. Its peak usually occurs when the model strikes a balance between correctly identifying positives (precision) and capturing all positives (recall). Hence, accuracy might attain its highest scores at threshold values somewhere in the middle, reflecting a balance between the trade-offs of precision and recall.

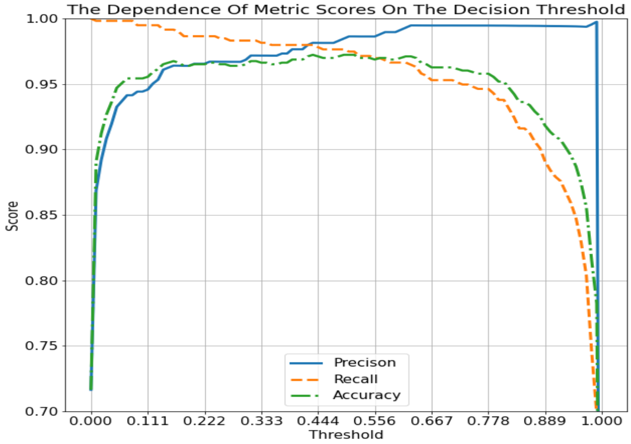


Figure 13: Score Vs Threshold of classification

This relationship between threshold, precision, recall, and accuracy underscores the importance of understanding the interplay among these metrics and how they fluctuate based on the decision threshold. Adjusting the threshold allows for a nuanced control over the model's behavior, catering to specific needs or priorities of the classification task at hand. Best threshold found at 0.44 and accuracy at 97% (Figure 13).

5. CONCLUSION

The impact of employing various regularization techniques, including image augmentation, dropout, and learning rate decay, within the CNN architecture is vividly demonstrated. Comparing the regularized CNN to the non-regularized or Base CNN, noticeable enhancements are evident. The regularized CNN displays superior performance, achieving a substantial maximum validation accuracy of 97% over 20 epochs. Particularly noteworthy is the absence of overfitting, indicated by the consistent alignment between training and validation accuracy. This absence suggests the potential for further accuracy gains with additional epochs. The robust performance of the regularized CNN underscores its effectiveness in preventing overfitting and enhancing the model's ability to generalize to unseen data, validating the success of incorporating various regularization techniques.

The precision, recall, and F1-score metrics, depicted in the confusion matrix (Fig. 11) and summarized in Table I, offer an in-depth evaluation of the classification model's performance across Class 0 and Class 1. With Class 0 exhibiting a precision of 94% and Class 1 reaching a precision of 99%, the model showcases high accuracy in classifying instances for both categories. Similarly, the recall rates of 96% for Class 0 and 98% for Class 1 demonstrate the model's ability to identify the majority of instances belonging to each class. The F1-scores further validate the overall performance, with Class 0 achieving a robust score of 0.95 and Class 1 demonstrating even higher performance at 0.98, affirming the model's accuracy, sensitivity, and balanced classification ability across both classes.This nuanced control over the model's behavior underscores the importance of understanding metric interplay and threshold adjustments for tailored classification performance, with the optimal threshold identified at 0.44, achieving a 97% accuracy.

REFERENCES

[1] G. Litjens, T. Kooi, B.E. Bejnordi, A .A .A . Setio, F. Ciompi, M. Ghafoorian, J.A. Van Der Laak, B. Van Ginneken, C.I. Sánchez, A survey on deep learning in medical image analysis, Med. Image Anal. 42 (2017) 60–88 .

[2] M. Xu, S. Yoon, A. Fuentes, J. Yang, D. Park, Style-consistent image translation: a novel data augmentation paradigm to improve plant disease recognition, Front. Plant Sci. 12: 773142. doi: 10.3389/fpls (2022) .

[3] S.C. Wong, A. Gatt, V. Stamatescu, M.D. McDonnell, Understanding data aug- mentation for classification: when to warp? in: 2016 international conference on digital image computing: techniques and applications (DICTA), IEEE, 2016, pp. 1–6 .

[4] L. Taylor, G. Nitschke, Improving deep learning with generic data augmentation, in: 2018 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, 2018, pp. 1542–1547

[5] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, Adv. Neural Inf. Process. Syst. 25 (2012) .

[6] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE conference on computer vision and pattern recog- nition, 2016, pp. 770–778 .

[7] Boyi Li, Felix Wu, Ser-Nam Lim, Serge Belongie, and Kilian Q. Weinberger. On feature normalization and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12383–12392, June 2021.

[8] Cherry Khosla and Baljit Singh Saini. Enhancing performance of deep learning models with different data augmentation techniques: A survey. In *2020 International Conference on Intelligent Engineering and Management (ICIEM)*, pages 79–85, 2020

[9] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data*, 6(1):1–48, 2019.

[10] Keyu Tian, Chen Lin, Ming Sun, Luping Zhou, Junjie Yan, and Wanli Ouyang. Improving autoaugment via augmentation-wise weight sharing. *arXiv preprint arXiv:2009.14737*, 2020.

[11] Chia-Wen Kuo, Chih-Yao Ma, Jia-Bin Huang, and Zsolt Kira. Featmatch: Feature-based augmentation for semi-supervised learning. In *European Conference on Computer Vision*, pages 479–495. Springer, 2020.

[12] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li and L. Fei-Fei, ImageNet: A Large-Scale Hierarchical Image Database. *IEEE Computer Vision and Pattern Recognition* (CVPR), 2009.

[13] Zheng, Y., Huang, J., Chen, T., Ou, Y., & Zhou, W. (2021). Transfer of Learning in the Convolutional Neural Networks on Classifying Geometric Shapes Based on Local or Global Invariants. *Frontiers in computational neuroscience*, *15*, 637144.

[14] Wang, X., Peng, Y., Lu, L., Lu, Z., Bagheri, M., & Summers, R. M. (2017). Chestx-ray8: Hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2097-2106).

[15] Kermany, D. S., Goldbaum, M., Cai, W., Valentim, C. C., Liang, H., Baxter, S. L., ... & Zhang, K. (2018). Identifying medical diagnoses and treatable diseases by image-based deep learning. cell, 172(5), 1122-1131.

[16] Islam, M. T., Aowal, M. A., Minhaz, A. T., & Ashraf, K. (2017). Abnormality detection and localization in chest x-rays using deep convolutional neural networks. arXiv preprint arXiv:1705.09850.