International Journal of All Research Education and Scientific Methods (IJARESM),
ISSN: 2455-6211, Volume 11, Issue 7, July-2023, Available online at: www.ijaresm.com

# Transforming Healthcare: A Paradigm Shift through AI-Smart Diagnosis

**Vishal Pattar[1], Satish Sharma[2], Tanishk Patil[3], Gauri Raykar[4], Piyush Patil[5], Laxman Pajai[6], Supriya Balote[7], Shraddha Pandit[8], Mayuri Thorat[9]**

[1,2,3,4,5,6,9]Student, Department of Artificial Intelligence and Machine Learning, PES's Modern College of Engineering, Pune, Maharashtra, India
[7]Assistant Professor, Department of Artificial Intelligence and Machine Learning, PES's Modern College of Engineering, Pune, Maharashtra, India
[8]Head of Department, Department of Artificial Intelligence and Machine Learning, PES's Modern Collegeof Engineering, Pune, Maharashtra, India

**********

## ABSTRACT

**This Research paper investigates the application of Deep Learning to analyze and interpret medical imaging data, such as CT scans, X-rays. The objective is to enhance medical diagnosis through image processing and Deep Learning Algorithms like Convolutional Neural Networks (CNN). The development of this model involved training using popular Python libraries such as TensorFlow and Keras. This research paper utilizes image datasets obtained from Kaggle for analysis purpose. For training purposes, an extensive dataset comprising more than 30,000 images was employed. The model is put into action by deploying it on a Telegram bot.**

**Keywords: AI, Bot, CNN, CT-scan, Deep Learning, Image Processing, Machine Learning, Medical Diagnosis, Python, TensorFlow.**

## INTRODUCTION

The primary objective of our system is to leverage AI technology effectively within the domain of healthcare, thereby facilitating its valuable application and contribution to the healthcare domain. This model harnesses the power of convolutional neural networks (CNNs) to classify medical images with remarkable precision, particularly focusing on Kidney, Brain-Tumor, Lungs, and Tuberculosis diseases.

Accurate and timely diagnosis plays a crucial role in the effective management of diseases and the subsequent improvement of patient outcomes. Traditional diagnostic approaches heavily rely on the expertise and experience of medical professionals, often leading to variability and subjectivity in diagnoses. This model aims to overcome these limitations by utilizing advanced deep learning techniques and a comprehensive dataset of more than 40,000 medical images.[1]

The CNN architecture serves as the backbone of this model, allowing it to extract intricate features and patterns from the medical images. By training the model on a vast dataset, the system learns to discern subtle variations, enabling it to accurately classify the presence or absence of Kidney, Brain-Tumor, Lungs, and Tuberculosis diseases. The utilization of these type of models has the potential to significantly augment the diagnostic process, enabling rapid and reliable assessments, early detection of diseases, and facilitating timely interventions.

This research paper aims to delve deeper into the technical aspects of this model, elucidating the underlying principles of CNN architecture, the process of dataset acquisition and preparation, and the training methodology employed.

## MODEL ARCHITECTURE

### A. General Model

The general model of an Artificial Neural Network (ANN) has a single input layer and one output layer, along with several hidden layers. Equation (1) shows how each neuron in the network applies a function F to an input vector X to produce an output vector Y.

$$F(X, W) = Y \quad (1)$$

The intensity of connections between neurons in nearby layers is represented by the weight vector W. When classifying photos using the weight vector, contextual information like the image shape frequently produces better results than pixel-based classification.[4]

International Journal of All Research Education and Scientific Methods (IJARESM),
ISSN: 2455-6211, Volume 11, Issue 7, July-2023, Available online at: www.ijaresm.com

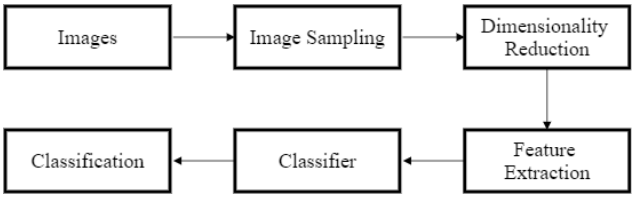



**Fig 2.1 Elementary Constitutes of CNN**

A popular model for utilizing contextual information is the Convolutional Neural Network (CNN), which is composed of four components: the convolution layer, pooling layer, fully connected layer and activation function. The description of each component's functionality is depicted in Fig 2.1 [6].

### B. Convolution Layer

A crucial element within convolutional neural networks (CNNs) is the convolutional layer, responsible for extracting features from input data such as images. The layer utilizes convolutional kernels to perform convolution operations on the input data and extract valuable features to pass it to next layer. This layer operates by convolving kernels/filters across the input data of images and performing the dot product between each kernel/filter and its corresponding patch of input. This process helps capture local spatial patterns and detect relevant features. As a result of the convolutional layer's operations, it generates a feature map that highlights the presence of various features across the input data.[3]

The formula for the convolution operation between a filter (kernel) and an input patch is as follows

$$C(x, y) = \sum_m \sum_n I(x + m, y + n).K(m, n) \quad (2)$$

Where,

C(x, y) represent the output value at position (x, y) in the feature map.
I(x + m, y + n) denotes the input value at position (x + m, y + n).
K(m, n) refers to the filter weight at position (m, n).

Some commonly used convolutional layers in convolutional neural networks (CNNs) include

1) *Conv2D* - This conventional 2D convolutional layer uses a series of trainable filters to execute convolution operations on 2D input feature maps.
2) *Dilated Convolution* - Also known as atrous convolution, it introduces gaps or "holes" in the filter to expand the receptive field and capture larger context without increasing the number of parameters.
3) *Transposed Convolution (Deconvolution)* - It is used in up sampling or "deconvolutional" operations, expanding the spatial dimensions of the input feature maps.
4) *Grouped Convolution* - It divides the input and filters into groups, enabling parallel processing and reducing the computational complexity.
5) *Depth wise Convolution* - It performs separate convolutions for each input channel using a dedicated kernel, reducing computation while capturing spatial correlations.

### C. Pooling Layer

The pooling layer is a component that appears after the convolutional layer when convolutional neural networks are constructed. It is designed with the purpose of down sampling the spatial dimensions of the input feature maps so that the size of the maps can be reduced while the information retained is deemed important.[7]

In pooling, the input feature map is partitioned into non-overlapping regions (often squares), and then an aggregation function is applied within each region. Pooling is most often done with max pooling, where the maximum value within each region is used to represent the region. Among the other pooling techniques, average pooling determines the average value within each region.[7]

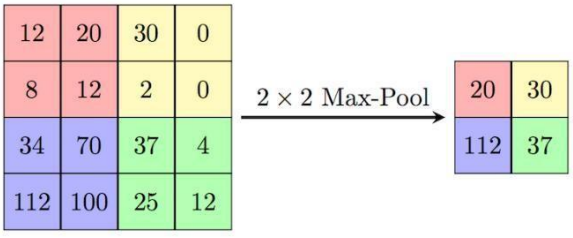



**Fig 2.2 MaxPooling2D Layer in CNN**

In the Fig 2.2 a 2x2 max pooling with a stride of 2 is applied to the input. Each 2x2 region in the input is reduced to a single value by selecting the maximum value within that region. The resulting output has a reduced spatial dimension but retains the most prominent features.[7]

Pooling layers, with their down sampling and aggregation operations, play a vital role in reducing the spatial dimensions while retaining relevant information in CNNs. They contribute to the hierarchical feature learning process and help extract high-level features for subsequent layers in the network.[7]

Some commonly used pooling layers in convolutional neural networks (CNNs) include

1) *Max Pooling* - Using this technique, the maximum value is determined within each non-overlapping rectangular section from the input feature map that has been divided into segments. The outcome is a reduction in the spatial dimensions while maintaining the key characteristics.
2) *Average Pooling* - This method downscales the spatial dimensions while maintaining the overall shape information by dividing the input feature map into non-overlapping sections and computing the average value within each region.
3) *Global Average Pooling* - It works by calculating the average value across the complete feature map, a single value per feature map is obtained, effectively reducing the spatial dimensions to 1x1.
4) *Sum Pooling* - It calculates the sum of values within each pooling region, providing a more holistic representation of the input feature map.
5) *Adaptive Pooling* - It dynamically adjusts the pooling regions' size based on the input feature map's dimensions, allowing for flexible down sampling regardless of input size.

The pooling layers have a vital role in decreasing the spatial dimensions of the feature maps, summarizing information, and improving computational efficiency. They serve the purpose of capturing the most essential features while discarding less significant spatial details, enabling the CNN to learn and generalize from the extracted features effectively.

### D. Fully Connected Layer

Fully connected layers in CNNs combine high-level abstract features from earlier layers, taking a flattened 1D vector as input. Each neuron is connected to every neuron in the previous layer with corresponding weights. Activation functions introduce non-linearity to learn complex patterns. The final fully connected layer acts as the output layer for predictions, and SoftMax is often used for class probabilities. During training, backpropagation adjusts weights to minimize prediction errors. Fully connected layers play a critical role in combining extracted features and enabling CNNs to excel in various tasks like image classification, object detection, and more.

Convolutional neural networks (CNNs) frequently use fully linked layers that include:

1) *Standard Fully Connected Layer* - Each neuron in the conventional fully connected layer is linked to every neuron in the layer above it, creating a completely connected network structure.
2) *Multi-Layer Perceptron (MLP)* - An MLP consists of a series of completely connected layers with non-linear activation functions interspersed. It gives the network more depth and nonlinearity, enabling more complicated representations and decision limits.
3) *Output Layer* - The final predictions or regression results are delivered by the output layer, a specialized fully linked layer. Depending on the job at hand, the output layer's neuron count changes.

### E. Activation Function

An important element that comes after the convolutional and pooling layers in convolutional neural networks (CNNs) is the activation function layer. The main goal of adding non-linear activation functions to the network is to bring about non-linearity, which will allow the network to successfully learn complex correlations between input features.

Activation functions apply a mathematical transformation to each neuron's output in the network, introducing non-linear properties that allow CNNs to model highly nonlinear relationships present in real-world data. Without activation functions, the network would effectively be a linear model, limiting its capacity to learn and represent complex patterns.[2]

There are several popular activation functions used in CNNs, including

1) *Rectified Linear Unit (ReLU)* - ReLU stands as one of the widely adopted activation functions in neural networks. It transforms all negative values to zero while preserving positive values. Mathematically, ReLU can be defined as follows

$$f(z) = \max(0, z) \tag{3}$$

ReLU offers computational efficiency and helps with gradient propagation during training.

2) *Sigmoid* - The sigmoid function compresses input values within the range of 0 and 1, making them interpretable as probabilities. It is characterized by a smooth, S-shaped curve and can be mathematically defined as follows

$$f(z) = \frac{1}{(1+e^{(-z)})} \quad (4)$$

Sigmoid functions are used when the output needs to be in a bounded range, such as binary classification tasks.

3) *Hyperbolic tangent (Tanh)* - The tanh function, similar to the sigmoid function, transforms the output values to lie between -1 and 1. It is mathematically defined as follows

$$f(z) = \frac{(e^z e^{(-z)})}{(e^z + e^{(-z)})} \quad (5)$$

Tanh is useful when the input data is centered around zero.

4) *Leaky ReLU* - Leaky ReLU is a variant of the ReLU function that introduces a small positive slope for negative input values. It addresses the issue of "dying ReLU" where certain neurons can become unresponsive during training due to a significant negative gradient. Mathematically, the leaky ReLU function is defined as follows

$$f(z) = \max(az, z) \quad (6)$$

where a is a small positive constant.

The selection of an activation function is contingent upon the particular task at hand, the architecture of the network, and the characteristics of the data. It is common to use ReLU or its variants as the default choice for most CNN architectures due to their computational efficiency and effectiveness in handling gradient propagation.

The activation function layer plays a vital role in introducing non-linearity and enabling CNNs to learn complex representations from input data. It adds the ability to model intricate patterns and significantly contributes to the network's capacity to make accurate predictions.

## METHODOLOGY

The methodology used for classifying images for diseases using a Convolutional Neural Network (CNN) typically involves the following steps

### A. Data Collection and Analyzing the data

A comprehensive dataset comprising more than 40,000 medical images, including CT-Scan, MRI Scan, and X-Ray images, specifically targeting diseases of interest such as Kidney, Brain-Tumor, Lungs, and Tuberculosis, was meticulously collected from Kaggle.

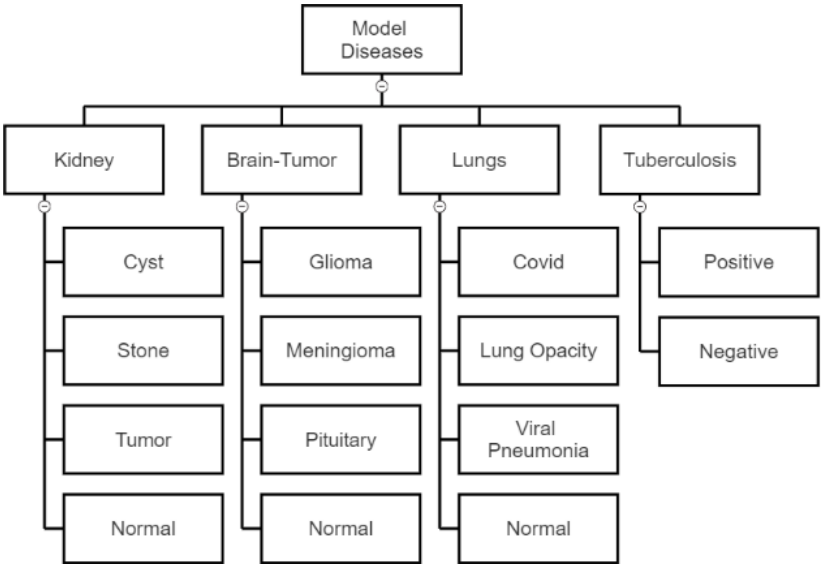



Fig 3.1 Diseases used for Classification

In Figure 3.1, a visual representation is provided, depicting the sub-types associated with each disease for classification purposes. The kidney diseases consist of cysts, stones, tumors, and normal cases. Similarly, the brain-tumor diseases include gliomas, meningiomas, pituitary tumors, and normal cases. The lung diseases encompass COVID-19, lung opacity, viral

pneumonia, and normal cases. Additionally, tuberculosis is categorized as either or negative, representing a binary classification task. It is worth noting that kidney, brain-tumor, and lung diseases fall under categorical classification, while tuberculosis is classified as binary.

### A. Image Processing

Image processing holds significant importance in the realm of deep learning, particularly in the context of convolutional neural network (CNN) models. It enables the extraction and manipulation of visual information from images, allowing the models to understand and interpret visual data effectively. In order to enhance the quality and utility of the medical images, several Image Processing techniques were applied, including Noise Removal, Segmentation, Padding, Contrast Adjustment, and more.

In the context of Brain-Tumor Disease, image processing involves pre-processing steps like noise removal and enhancement, followed by segmentation to isolate the tumor region. Post-processing techniques refine the boundaries, and quantitative features are extracted for classification and analysis, enabling accurate diagnosis and treatment planning.

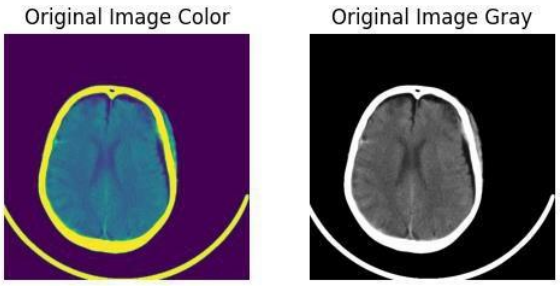



**Fig 3.2 Original Image without processing**

Figure 3.2 displays the original CT-Scan image of a Brain-Tumor obtained from a patient, which exhibits certain elements that hinder the overall efficiency of the model. Notably, there is noise present around the head region, inadequate padding, slight tilting, and non-compliance with the prescribed size. To address these issues, image processing techniques can be applied to rectify these errors and enhance the image quality.

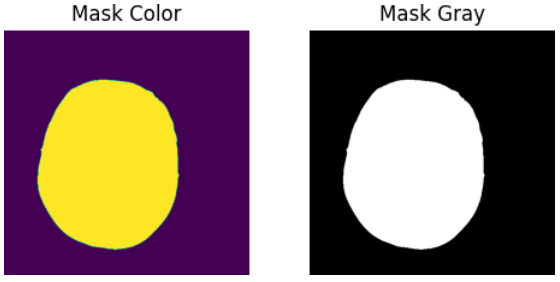



**Fig 3.3 Masking image for further processing**

Figure 3.3 illustrates the application of masking on the image. Masking is used in image processing to selectively modify or extract specific regions of an image. It enables focusing on areas of interest, separating objects from backgrounds, performing precise edits, and aiding in tasks like segmentation and windowing.[9]

Following the completion of masking, the subsequent step involves the detection and correction of any tilting present in the image. This is achieved by identifying the end points of the masked image on each side and locating the largest contour line, which is then used to determine the required tilt adjustment for the entire image. Furthermore, additional image processing techniques such as blurring and enhancement are applied to further enhance the quality of the image for subsequent processing stages.

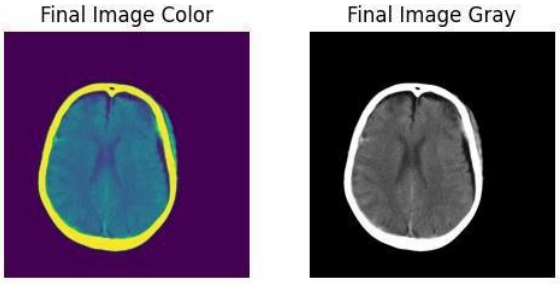



**Fig 3.4 Final Image after pre-processing**

Figure 3.4 showcases the successful completion of the pre-processing steps, resulting in a prepared image. However, it is evident that there is unnecessary background present, which needs to be removed. Additionally, the image is not centered properly, requiring adjustments to ensure proper alignment.

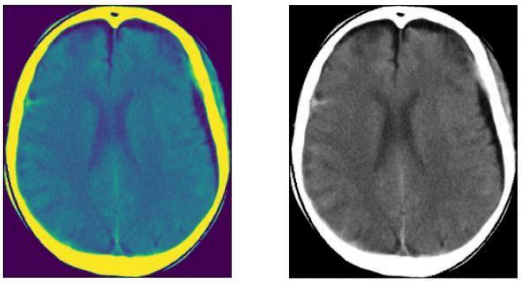


**Fig 3.5 Segmenting the image**

In Figure 3.5, the image segmentation process has been successfully carried out. This involved identifying the extreme end points of the image on each side and creating a rectangle around the relevant portion, while disregarding the remaining areas. However, the resulting image does not meet the targeted size of (512px, 512px) in terms of width and height.

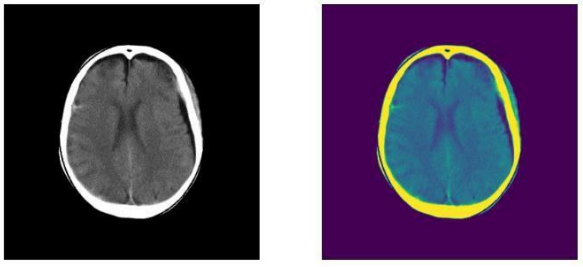


**Fig 3.6 Final Image after post-processing**

Figure 3.6 demonstrates the addition of padding to the remaining portion of the image, ensuring the maintenance of its spatial dimensions. Padding involves the inclusion of extra pixels, often with zero values, along the image borders, thereby increasing its size or preserving spatial dimensions during convolutional operations. The final step involves converting the image into grayscale, transforming it into a black and white representation, facilitating computational processing. Finally, at this stage, the image is prepared and ready for training purpose.

### B. Model Architecture Design

In situations where the available training images are limited or when there is a need to enhance the diversity of the datasets, the approach of data augmentation is employed. Data augmentation is a widely employed technique in deep learning that aims to artificially enhance the size and diversity of a training dataset. It accomplishes this by applying diverse transformations and modifications to the existing data.

Data augmentation can involve a range of transformations, such as rotation, scaling, translation, flipping, cropping, adding noise, changing brightness/contrast, or applying geometric distortions. By generating new samples with these transformations, data augmentation reduces overfitting, enhances the model's ability to handle variations in the input data, and improves its performance on unseen data.[12]

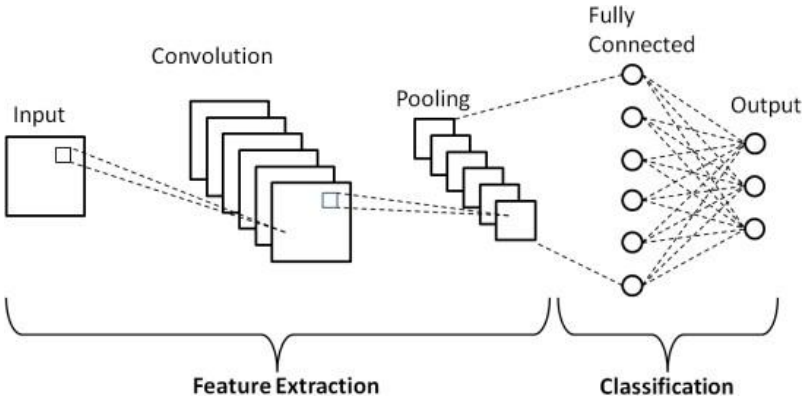



**Fig 3.7 CNN Architecture**

CNN architectures are designed according to the characteristics of medical images as well as the level of difficulty of the classification task. Several convolutional layers are layered on top of a pooling layer, followed by fully connected layers.

Layers, kernel sizes, and filters are selected according to the classification task's requirements.

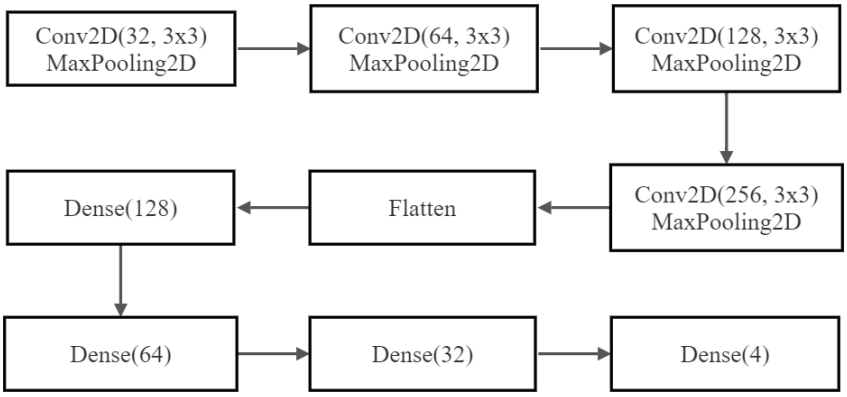



**Fig 3.8 CNN Layers used for Model Designing**

Figure 3.8 presents a comprehensive illustration of the layers used in the model design. The initial stages consist of conv2D layers, followed by pooling layers, and ultimately concluding with fully connected dense layers. To be more precise, a conv2D layer with a kernel/filter of size 3x3 and 32 filters is applied. Subsequently, a MaxPooling2D layer is used for dimensionality reduction and retaining relevant information. This process is repeated with additional convolutional and pooling layers, gradually increasing the number of filters (64, 128, 256) while maintaining the same attributes. To put it simply, the max-pooling layers ensure that the convolution layers valuable information is retained as the convolution layers extract distinctive features from the image.[7]

The output is placed through a Flatten layer after the convolutional and pooling layers. The Flatten layer transforms the multidimensional output into a one-dimensional vector, thereby facilitating the connection with fully connected layers. Its role is to gather the characteristic features extracted from each convolutional and pooling layer and consolidate them into a single layer. Convolutional and pooling layers are connected by the Flatten layer, which essentially serves as a link to the next fully connected layers.

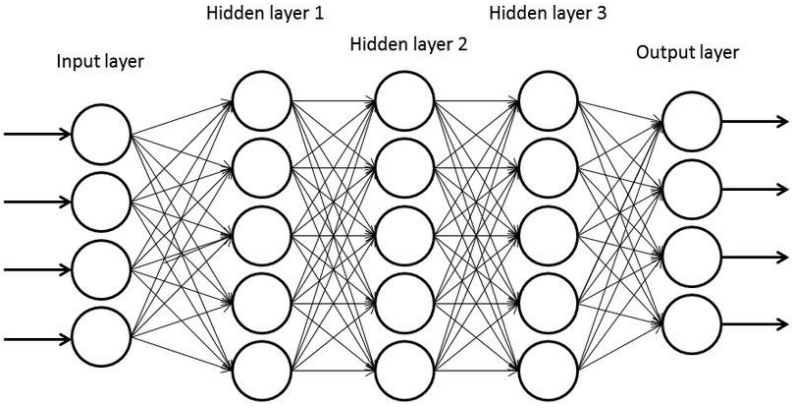



**Fig 3.9 Architecture of Standard Full Connected Layer**

Finally, we reach the fully connected layers where a standard fully connected layer (Dense Layer) is employed to implement the network structure. These layers learn intricate relationships and patterns from the extracted features and generate the final decision or output based on the learned representations. The number of neurons in these layers progressively decreases until it matches the number of categorical classes within the datasets. In this case, there are 4 sub-diseases present in each disease, resulting in 4 neurons, and 2 neurons for tuberculosis disease due to its binary classification nature.

The ReLU activation function was incorporated into our model for its simplicity and its ability to provide non-linearity and gradient propagation. ReLU effectively addresses the vanishing gradient problem in CNNs, enhances network training, and aids in faster convergence. By setting negative values to zero, ReLU introduces sparsity, facilitates efficient computation, and preserves positive activations to emphasize important features. Overall, ReLU activation contributes to faster training, improved network capacity, and superior performance in computer vision tasks.

### C. Training the CNN

The prepared image, post image processing, cannot be directly inputted into the system for analysis. Therefore, it is necessary to convert each image from the JPG format to a NumPy array, enabling the system to comprehend the image data. Additionally, all the pixel values are rescaled from the original range of 0-255 to the range of 0-1, as binary calculations can

be performed more efficiently within this normalized range.[6]

Several other parameters, including epoch, batch size, class mode, color mode, shuffle, loss type, optimizer, and metrics, need to be predetermined based on the type of classification and the complexity of the datasets.

A complete run through the entire training dataset is referred to as an epoch during the training phase of a neural network. The difficulty of the problem, the size of the dataset, and the available computing capacity must all be taken into account when deciding how many epochs to utilize.[12]

The number of training examples handled during one forward and backward pass (iteration) is referred to as the batch size during the training phase. There are a number of factors that determine the optimal batch size, such as the amount of memory available, the complexity of the model, and the size of the dataset.

It is important to note that epoch size and batch size influence the duration and convergence of the training. Class mode defines the type of classification, such as binary or categorical. Color mode determines the color representation of the images, such as grayscale or RGB. Shuffle determines whether the training data should be shuffled before each epoch. The choice of loss type and optimizer impacts the model's ability to minimize errors and optimize the learning process.

Utilizing the specified parameters, including the layers mentioned in Figure 3.8, the model is trained. The duration of the training process can vary depending on the system's hardware, ranging from a few minutes to several hours or even days. Once the training, or fitting, is completed, the trained model is saved in the .h5 format for future usage and reference. Saving the model allows for easy access and utilization in subsequent tasks or analysis.

The ".h5" format refers to the Hierarchical Data Format version 5. It is a file format commonly used in the realm of data science and machine learning to store and manage large amounts of data, particularly for deep learning models.

**D. Evaluation and Testing**

During the evaluation phase, the previously saved .h5 file containing the trained model is loaded into the system. Then, the system is provided with input images in the desired format, and the model predicts the corresponding output.

After the training is finished, the trained model's performance is assessed using an independent test set. Several measures, including accuracy, precision, recall, F1-score, and loss, are generated to evaluate the model's effectiveness in classifying diseases. These metrics provide insights into the model's capability to accurately classify diseases. Following the testing of the data, a confusion matrix is constructed based on the gathered results. This matrix is then utilized to evaluate the model's performance and calculate various metrics, including accuracy, loss, and more.

Confusion Matrix: To visualize and assess the performance of machine learning models in classification tasks, confusion matrix performance assessment metrics are utilized. The dataset's actual classes and anticipated classes are displayed in a square matrix. The rows in the matrix reflect the expected classes, and the columns in the matrix indicate the actual classes. Each cell in the matrix denotes the number or percentage of events that were correctly or wrongly categorized.[5]

Example of Confusion Matrix for Binary Classification:

| Number of samples | Predicted Positive | Predicted Negative |
|---|---|---|
| Actual Positive | True Positive (TP) | False Positive (FP) |
| Actual Negative | False Negative (FN) | True Negative (TN) |

Several terms that relate to the confusion matrix

1) *True Positives (TP)* - The number of times the model accurately predicted a positive class is referred to as TP. In other words, TP represents the situations where the model correctly identifies them as positive and the actual class is positive.
2) *True Negatives (TN)* - The number of times the model correctly predicts a negative class is represented by TN. It denotes the circumstances in which The model correctly recognizes them as negative since they actually belong to the negative class.
3) *False Positives (FP)* - The number of times the model predicts a positive class when it should not. It refers to

situations where the model incorrectly labels something as positive even though it actually belongs to a negative category (false positive).

4) *False Negatives (FN)* - The number of times the model incorrectly predicts a negative class is indicated by FN. It highlights the situations in which the true class is positive but the model mistakenly labels them as negative (false negative).

Here are some commonly used model's evaluation metrics, their meanings, and the formulas to calculate them from a confusion matrix.

1) *Accuracy* - Accuracy is determined by assessing the correctness of the model's predictions, taking into account both true positives and true negatives. Accuracy can be calculated using the following formula, utilizing the values from the confusion matrix

$$\text{Accuracy} = \frac{\text{TN+TP}}{\text{TP+FN+FP+TN}} \tag{7}$$

2) *Recall (Sensitivity or True Positive Rate)* - The Recall of a model is the ability of the model to correctly identify positive instances (true positives) from the total number of positive instances.

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \tag{8}$$

3) *Precision* - This term describes how accurately the model is able to identify positive instances (true positives) from the total instances predicted as positive.

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{9}$$

4) *F1-Score* - The F1-Score provides a balanced measure of the model's performance by combining precision and recall into a single metric.

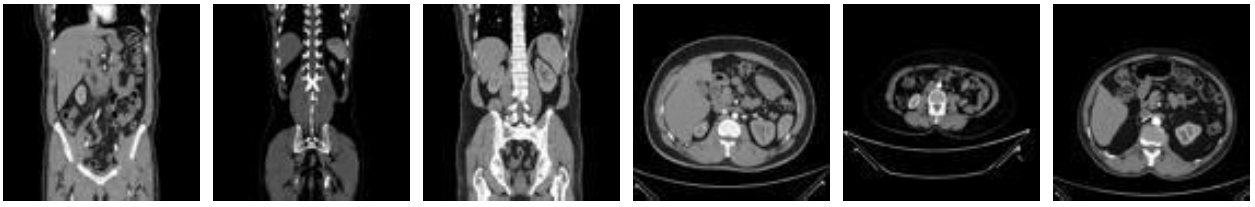
$$\text{F1} - \text{Score} = \frac{(2)*(\text{Precision})*(\text{Recall})}{(\text{Precision+Recall})} \tag{10}$$

5) *Loss* - Loss represents the measure of dissimilarity between the predicted output and the actual target output. The objective is to minimize this loss function during training to improve the accuracy of the model's predictions.

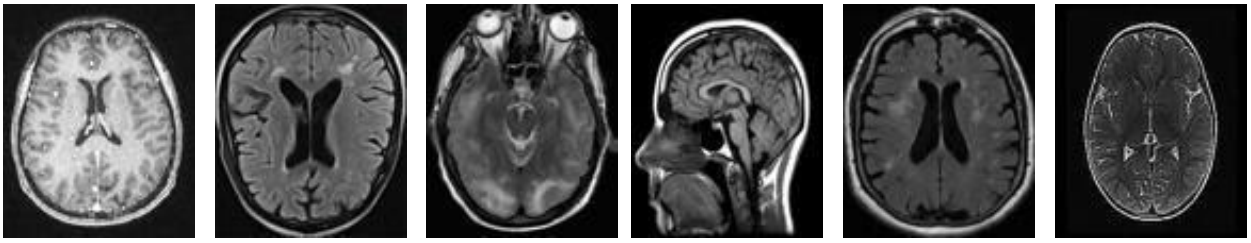
$$\text{Loss} = \frac{\text{FN+FP}}{\text{TP+FN+FP+TN}} \tag{11}$$

## SAMPLE IMAGE DATASETS

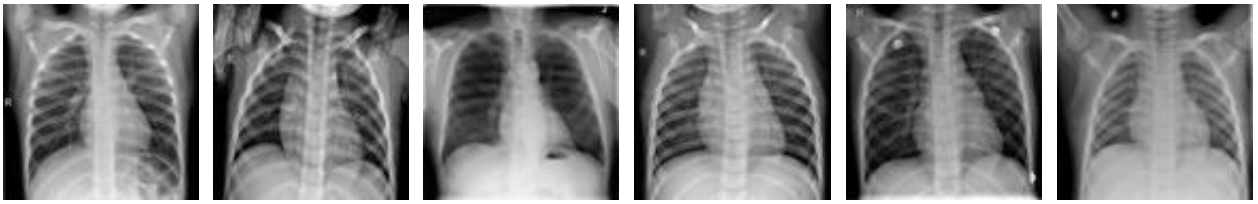
### A. Kidney disease dataset



### B. Brain-Tumor disease dataset



### C. Lung disease dataset



## DISEASES COMPREHENSION

### A. Kidney Disease

1) *Cyst* - Fluid-filled sacs that form in the kidney tissue, often without symptoms initially but can cause pain and urinary issues as they grow.[8]
2) *Stone* - Hard mineral and salt deposits in the kidneys that can cause severe pain, blood in the urine, and urinary tract symptoms.[8]
3) *Tumor* - An abnormal growth of cells in the kidney, which can be either benign or malignant, often causing symptoms like blood in the urine, pain in the side or back, and urinary tract infections.[8]

### B. Brain-Tumor Disease

1) *Glioma* - This condition affects the brain by causing abnormal cell growth, resulting in headaches, seizures, and cognitive decline.[9]
2) *Meningioma* - Brain tumor that arises from the meninges, the protective membranes surrounding the brain and spinal cord, often causing headaches, vision changes, and hormonal imbalances.[9]
3) *Pituitary* - Abnormal growth in the pituitary gland, leading to hormone imbalances, headaches, vision problems, and other symptoms.[9]

### C. Lung Disease

1) *Covid-19* - Respiratory illness caused by the SARS-CoV-2 virus, with symptoms including fever, cough, loss of taste or smell, and in severe cases, respiratory distress.[10]
2) *Lung Opacity* - Abnormal density or opacity seen in lung imaging, indicating various conditions affecting the lungs such as pneumonia, infections, or other lung diseases.[10]
3) *Viral Pneumonia* - Lung infection caused by viral agents, characterized by symptoms such as cough, fever, difficulty breathing, and fatigue.[10]

**D. Tuberculosis** - Contagious bacterial infection primarily affecting the lungs, causing symptoms such as persistent cough, weight loss, fatigue, and night sweats.[11]

## HELPFUL HINTS

### A. Abbreviations and Acronyms

**Table 1.1**

| Sr no. | Acronyms | | Abbreviations |
|---|---|---|---|
| 1 | AI | — | Artificial Intelligence |
| 2 | ANN | — | Artificial Neural Network |
| 3 | CNN | — | Convolution Neural Network |
| 4 | DL | — | Deep Learning |
| 5 | FN | — | False Negative |
| 6 | FP | — | False Positive |
| 7 | ReLU | — | Rectified Linear Unit |
| 8 | TB | — | Tuberculosis |
| 9 | GB | — | Giga Bytes |
| 10 | RAM | — | Random Access Memory |
| 11 | TN | — | True Negative |
| 12 | TP | — | True Positive |

### B. Numerical Analysis of Datasets

**Table 1.2**

| Diseases | Sub-diseases | No of images used for training |
|---|---|---|
| Kidney | Cyst | 3703 |
| | Stone | 1377 |
| | Tumor | 2283 |
| | Normal | 5077 |
| Brain-Tumor | Glioma | 1296 |
| | Meningioma | 1316 |
| | Pituitary | 1405 |
| | Normal | 1600 |
| Lungs | Covid-19 | 3616 |
| | Lung Opacity | 6012 |
| | Viral Pneumonia | 1345 |
| | Normal | 10192 |
| Tuberculosis | Positive case | 630 |
| | Negative case | 1305 |
| Total count | 10 sub-diseases | 41,156 |

### C. Resources

1) *Telegram bot Link* - Available: https://t.me/AI_Smart_Diagnosis_Bot
2) *Working model Demo* - Available: https://tinyurl.com/25w4782m

### D. Dataset Sources

1) *Kidney Disease* - Available: https://www.kaggle.com/datasets/nazmul0087/ct-kidney-dataset-normal-cyst-tumor-and-stone
2) *Brain-Tumor Disease* - Available: https://www.kaggle.com/datasets/muhammadd7/brain-tumor-mri-images
3) *Lung Disease* - Available: https://www.kaggle.com/datasets/tawsifurrahman/covid19-radiography-database

## RESULTS AND METRICES

Figures 6.1, 6.2, and 6.3 present graphs displaying various model evaluation metrics, including accuracy, precision, recall, and loss, for different diseases, namely kidney, brain-tumor, and lung diseases. The X-axis represents the number of epochs utilized during model training, while the Y-axis depicts the evaluation metrics.
For all three figures, the blue line corresponds to the Training data, while the orange line represents the validation data. The computation of these metrics during the training process enabled tracking of overfitting and underfitting tendencies, using

50 epochs for each model.

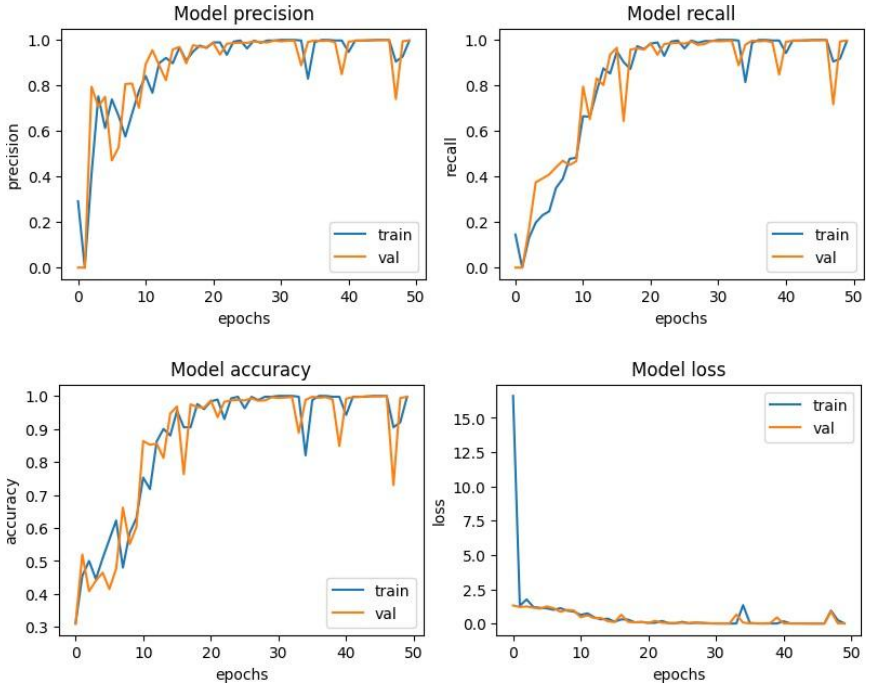



**Fig 6.1 Model metrics for kidney diseases**

In Fig 6.1, which shows the testing data for kidney disease, the graph suggests the presence of some overfitting due to limited diversity in the images and minimal gap between the validation and training data. However, overall, the model does not exhibit any underfit condition and is closer to a just-fit state.

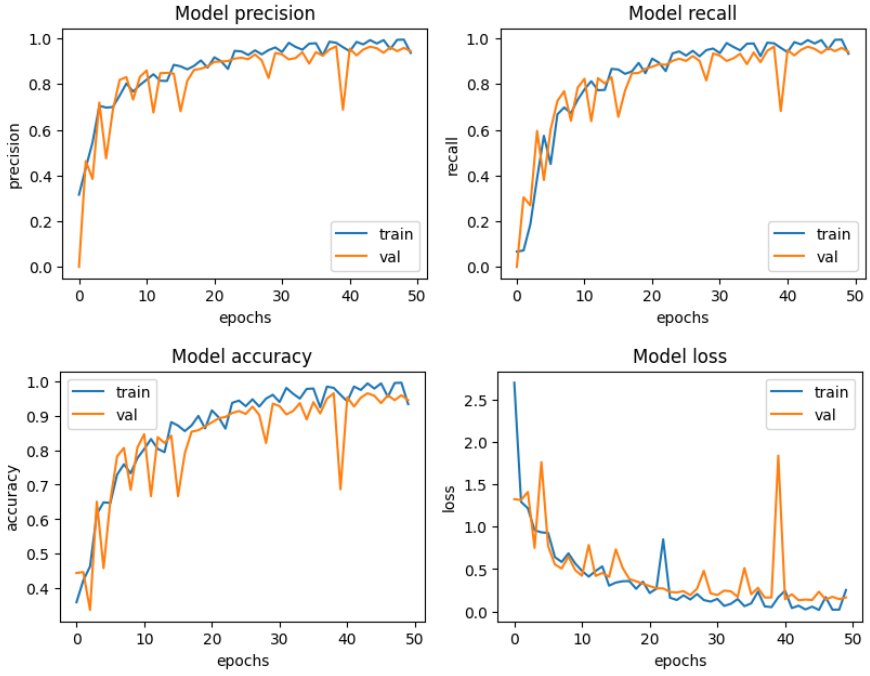



**Fig 6.2 Model metrics for brain-tumor diseases**

Fig 6.2 displays the testing data for brain-tumor disease. The graph indicates that the model shows no signs of overfitting or underfitting, as there is a noticeable gap or divergence between the validation data and training data. Overall, the model is in a nearly just-fit condition.

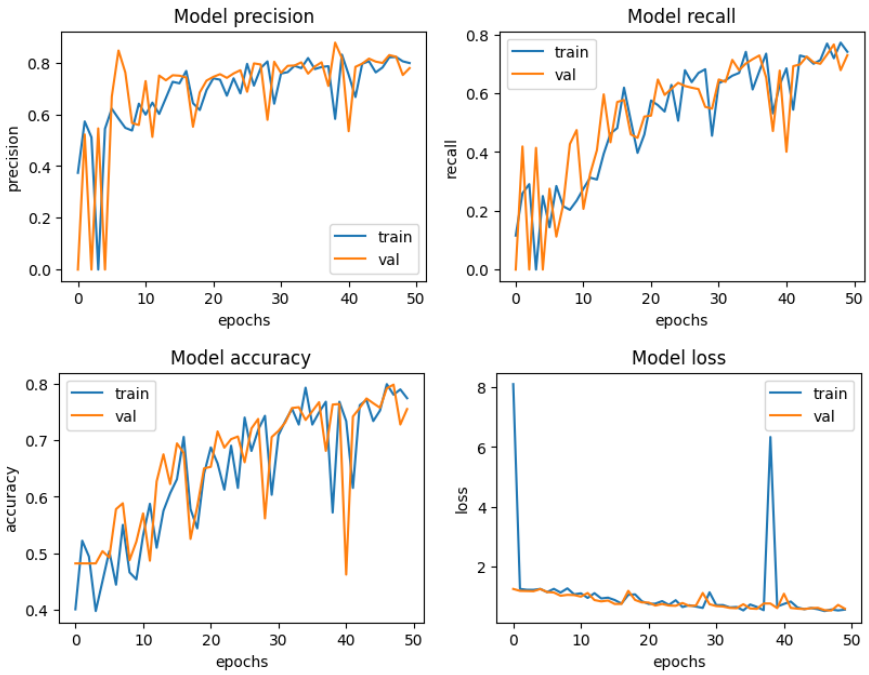



**Fig 6.3 Model metrics for lung diseases**

In Fig 6.3, representing the testing data for lung disease, the graph suggests some underfitting due to the utilization of poor datasets during training. Nonetheless, the model displays good accuracy despite the limitations of the datasets. These comprehensive evaluations of the model's performance across different diseases provide valuable insights into its behavior and generalization capabilities.

**Table 1.3**
**Metrics evaluated for test data**

| Parameters/Diseases | Kidney (%) | Brain-Tumor (%) | Lungs (%) | Tuberculosis (%) |
|---|---|---|---|---|
| **Accuracy** | 98.76 | 94.88 | 75.93 | 85.08 |
| **Precision** | 98.48 | 94.72 | 83.00 | 85.06 |
| **Recall** | 98.36 | 94.76 | 71.20 | 85.04 |
| **F1-Score** | 98.42 | 92.73 | 75.35 | 85.05 |
| **Loss** | 1.24 | 5.12 | 24.07 | 14.92 |

The deep learning model utilized for classifying medical images has shown promising results. With high accuracy and precision, the model effectively distinguishes between different medical conditions and accurately classifies the images into relevant categories. It demonstrates the potential for assisting healthcare professionals in diagnosing diseases and conditions, enabling timely and accurate treatment. The model's performance demonstrates how deep learning techniques can benefit medical image analysis, resulting in better patient care and diagnostic accuracy.[7]

Furthermore, the deep learning model exhibits robustness and generalizability, performing consistently well across diverse datasets and varying imaging modalities. Its ability to handle complex patterns and subtle features within medical images enhances its diagnostic capabilities. The model's efficiency in processing large volumes of data allows for faster and automated analysis, aiding in reducing the burden on medical professionals and enabling faster decision-making. The impressive results obtained by the deep learning model underline its potential as a valuable tool in medical imaging for improving accuracy, efficiency, and patient outcomes.

## CONCLUSION

Overall, this model is an innovative development in healthcare that improves medical decision-making and enhances patient care. By harnessing artificial intelligence, it rapidly processes vast patient data, leading to faster and more accurate diagnoses. AI algorithms identify patterns and correlations, reducing diagnostic errors and augmenting healthcare professionals' capabilities. This integration addresses disparities and improves access to care through remote diagnosis and telemedicine. However, ethical considerations, privacy concerns, and regulatory frameworks must be addressed to ensure responsible data

use. Proper training for healthcare professionals is crucial for effective utilization of AI systems. In conclusion, this model could transform healthcare, improving accuracy, personalized medicine, and patient outcomes while bridging gaps and promoting equal access to quality care.

## REFERENCES

[1] McRae, Michael P., Kritika S. Rajsri, Timothy M. Alcorn, and John T. McDevitt. 2022. "Smart Diagnostics: Combining Artificial Intelligence and In Vitro Diagnostics" Sensors 22, no. 17: 6355, https://doi.org/10.3390/s22176355.

[2] H. -C. Shin et al., "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning," in IEEE Transactions on Medical Imaging, vol. 35, no. 5, pp. 1285-1298, May 2016, https://doi.org/10.1109/TMI.2016.2528162.

[3] W. Hao, W. Yizhou, L. Yaqin and S. Zhili, "The Role of Activation Function in CNN," 2020 2nd International Conference on Information Technology and Computer Application (ITCA), Guangzhou, China, 2020, pp. 429- 432, https://doi.org/10.1109/ITCA52113.2020.00096.

[4] S Agatonovic-Kustrin, R Beresford, Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research, Journal of Pharmaceutical and Biomedical Analysis, Volume 22, Issue 5, 2000, Pages 717-727, ISSN 0731-7085, https://doi.org/10.1016/S0731-7085(99)00272-1.

[5] Yadong Wang, Yanlin Jia, Yuhang Tian, Jin Xiao, Deep reinforcement learning with the confusion-matrixbased dynamic reward function for customer credit scoring, Expert Systems with Applications, Volume 200, 2022, 117013, ISSN 0957-4174, https://doi.org/10.1016/j.eswa.2022.117013.

[6] Sakshi Indolia, Anil Kumar Goswami, S.P. Mishra, Pooja Asopa, Conceptual Understanding of Convolutional Neural Network- A Deep Learning Approach, Procedia Computer Science, Volume 132, 2018, Pages 679-688, ISSN 1877-0509, https://doi.org/10.1016/j.procs.2018.05.069

[7] Vincent Dumoulin, Francesco Visin - A guide to convolution arithmetic for deep learning (BibTeX), 2018, https://doi.org/10.48550/arXiv.1603.07285

[8] Snyder S, Pendergraph B. Detection and evaluation of chronic kidney disease. Am Fam Physician. 2005 Nov 1;72(9):1723-32. PMID: 16300034.

[9] M. Siar and M. Teshnehlab, "Brain Tumor Detection Using Deep Neural Network and Machine Learning Algorithm," 2019 9th International Conference on Computer and Knowledge Engineering International Journal of All Research Education and Scientific Methods (IJARESM), ISSN: 2455-6211, Volume 11, Issue 7, July-2023, Available online at: www.ijaresm.com Page | 2283 (ICCKE), Mashhad, Iran, 2019, pp. 363-368, https://doi.org/10.1109/ICCKE48569.2019.8964 846.

[10] Megan K. Dishop, Paediatric Interstitial Lung Disease: Classification and Definitions, Paediatric Respiratory Reviews, Volume 12, Issue 4, 2011, Pages 230-237, ISSN 1526-0542, https://doi.org/10.1016/j.prrv.2011.01.002.

[11] Sudre P, ten Dam G, Kochi A. Tuberculosis: a global overview of the situation today. Bull World Health Organ. 1992;70(2):149-59. PMID: 1600578; PMCID: PMC2393290.

[12] Anwar, S.M., Majid, M., Qayyum, A. et al. Medical Image Analysis using Convolutional Neural Networks: A Review. J Med Syst 42, 226 (2018), https://doi.org/10.1007/s10916-018- 1088-1

[13] X. Chen, Z. Chen, Z. Huang, W. Chen, and Y. Jin, "Artificial intelligence for diagnosis of lung cancer: a systematic review and meta-analysis," European Radiology, 2019.

[14] S. R. Patil and F. Deeba, "Implementation of Artificial Intelligence in Disease Prediction and Healthcare System- A Survey," in Innovations in Power and Advanced Computing Technologies (i-PACT), 2021, https://ieeexplore.ieee.org/document/9696698

[15] S. D. Raj and Karthiban, "Applications of Artificial Intelligence in Healthcare," in International Conference on Computer Communication and Informatics (ICCCI), 2022, https://ieeexplore.ieee.org/document/9741057 [17]. Tekkeşin Aİ. Artificial Intelligence in Healthcare: Past, Present and Future. Anatol J Cardiol. 2019 Oct;22(Suppl 2):8-9. doi: 10.14744/AnatolJCardiol.2019.28661. PMID: 31670713.