Classifying Chest X-Ray Images Using CNN Architectures

Costanza Fioroni
Department of Computer Engineering
Istanbul Technical University
fioroni23@itu.edu.tr
922310037

Antonio Spampinato
Department of Computer Engineering
Istanbul Technical University
spampinato23@itu.edu.tr
922310013

***Abstract*—The paper "Densely Connected Convolutional Networks" Huang et al. introduce the Dense Convolutional Network (DenseNet), which innovates by connecting each layer to every other in a feed-forward fashion, creating a densely connected model that alleviates the vanishing-gradient problem, strengthens feature propagation, encourages feature reuse, and significantly reduces the number of parameters [1]. This project aims to develop a deep learning model capable of classifying various diseases from chest X-ray images provided by an existing Kaggle Dataset. This will be achieved by using a pre-trained model of popular CNN architectures such as DenseNet and ResNet stated in the related article [1]. In addition to leveraging pre-trained models, the project will focus on refining these models through comprehensive fine-tuning and systematic hyperparameter optimization to improve diagnostic accuracy.**

I. Introduction

The advent of artificial intelligence (AI) and machine learning (ML) has revolutionized numerous fields, with computer vision (CV) standing out as a prime example of rapid advancement and application. This technology has transcended traditional data analysis, leveraging the power of visual data to enable machines to perform tasks that were once the exclusive domain of human vision. The implications of such advancements are particularly profound in the medical field, where the analysis of visual data such as X-ray images plays a critical role in diagnosis and patient care.

Recent years have seen a surge in research focused on the application of AI and ML to medical imaging, with a significant emphasis on the development of algorithms capable of diagnosing diseases from X-ray images. The potential of these technologies to support and improve the accuracy of medical diagnostics is immense, promising to deliver faster, more reliable, and accessible healthcare services.

This project is situated at the intersection of these developments, proposing a comparative analysis of DenseNet and ResNet, two important convolutional neural network (CNN) architectures, which represent the cutting edge in image recognition technology. The project will explore the efficacy of these models in classifying diseases from chest X-ray images, with a focus on fine-tuning, hyperparameter optimization, learning rate selection, and regularization to optimize performance.

The following document delves deeper into the project, outlining the objectives, anticipated impact, and novel contributions against the backdrop of current research. It also details the project's scope, including the work breakdown structure and other pertinent project information. Assumptions underlying the project's methodology and timeline are presented, leading up to a comprehensive list of deliverables and a detailed project schedule. By conducting this research, we aim to contribute to the ongoing conversation in the medical AI community and to provide insights that could shape future diagnostic procedures. The goal is to harness the potential of AI to improve outcomes for patients worldwide, making a tangible difference in the realm of medical care.

II. Project Description

In this project, chest X-ray images will be classified for different diseases such as: Abscess, Ards, Atelectasis, Cardiomegaly, Pneumonia, Tuberculosis, etc.

The dataset used in this project is a collection of 3 datasets from Kaggle platform:

- "Chest X-ray – 17 Diseases" [2], which including x-rays in 2 different formats JPEG and DCM and contains 17 folders corresponding to the name of the disease/condition.
- "Chest X-Ray Images (Pneumonia)" [3] that contains 5863 x-ray images in JPEG format which are categorized as pneumonia or normal.
- "Tuberculosis (TB) Chest X-ray Database" [4] contains CXR images of Normal (3500) and patients with TB (700 TB images in publicly accessible and 2800 TB images downloaded from NIAID TB portal [5] by signing an agreement).

A. Goals of Project

This project will have 2 main purposes:

- Classifying X-ray images for diseases
- Comparing two different CNN architectures (DenseNet and ResNet)

B. Impact of Solution

In this project, we will show difference between DenseNet and ResNet architectures with different hyperparameter settings, data augmentation techniques, and regularization methods. At the end of the project the results will be given in a comparative manner in the final report.

C. State-of-the-Art

The advancements in deep learning for image recognition have been significantly propelled by the introduction of DenseNet and ResNet, two architectures that have each set new benchmarks in the field. DenseNet, as introduced by Huang et al. [1], improves the flow of information and gradients throughout the network, which has been fundamental in mitigating the vanishing-gradient problem and has fostered more effective training of deep network architectures. In DenseNet, each layer receives feature maps from all preceding layers as inputs and passes its own feature maps to all subsequent layers in a feed-forward fashion, promoting feature reuse, reducing the number of parameters, and enhancing training efficiency. This unique feature is particularly advantageous in medical imaging diagnostics.

Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112 × 112	7 × 7 conv, stride 2			
Pooling	56 × 56	3 × 3 max pool, stride 2			
Dense Block (1)	56 × 56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56 × 56	1 × 1 conv			
	28 × 28	2 × 2 average pool, stride 2			
Dense Block (2)	28 × 28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28 × 28	1 × 1 conv			
	14 × 14	2 × 2 average pool, stride 2			
Dense Block (3)	14 × 14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14 × 14	1 × 1 conv			
	7 × 7	2 × 2 average pool, stride 2			
Dense Block (4)	7 × 7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1 × 1	7 × 7 global average pool			
		1000D fully-connected, softmax			

Fig. 1: DenseNet Architecture

ResNet, developed by He et al. [6], introduces a different approach to deep network training with its residual learning framework, enabling the training of networks that are substantially deeper than traditional models. ResNet's shortcut connections, or "skip connections" prevent the degradation problem by allowing unhindered information flow across layers. The result is a dramatic reduction in error rates, as demonstrated by its performance on ImageNet and other datasets. With its "bottleneck" design, ResNet efficiently manages the increase in depth, allowing for improved performance without a corresponding surge in computational demand.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix} \times 3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

ures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of block

Fig. 2: ResNet Architecture

Both architectures have set new standards for accuracy and efficiency in image recognition tasks, with DenseNet achieving state-of-the-art results on CIFAR datasets and ResNet achieving top results in ILSVRC and COCO challenges. The application of these models to medical imaging is a testament to their flexibility and robustness. DenseNet's parameter efficiency and compact feature growth, alongside ResNet's ability to train exceptionally deep networks without complexity penalties, offer a compelling toolkit for tackling the classification of diseases from chest X-ray images. In the proposed project, the pre-trained DenseNet and ResNet models will be fine-tuned and optimized to create a robust diagnostic tool. These models will be adapted to classify various diseases, with a focus on improving the accuracy and reliability of medical diagnostics. This comparative analysis will shed light on the adaptability and scalability of these architectures in a critical domain, setting a precedent for future applications in medical AI.

D. Novel Contributions

In this project, we will explore the effectiveness of CNN models, specifically DenseNet and ResNet, in diagnosing a variety of diseases from chest X-ray images. We aim to determine the performance benefits of these deep learning architectures by fine-tuning pre-trained models to our dataset, which comprises multiple diseases. The project will focus on evaluating and comparing the models based on performance, computational efficiency, and the clarity of their diagnostic outputs. This research is intended to advance the application of CNNs in medical image analysis, providing a deeper understanding of their adaptability and precision in clinical settings.

III. Applied Processes and Methods

This section includes information about the dataset, the used technologies, the applied processes and methods.

A. Dataset

The dataset used in the project is a compilation of three different datasets, collectively comprising eighteen distinct classes:

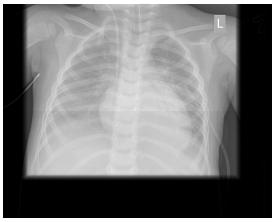
1) Normal
2) Pneumonia
3) Tuberculosis
4) Fracture
5) Emphysema
6) Hydrothorax
7) Post inflammatory changes
8) Hydropneumothorax
9) Ards
10) Scoliosis
11) Venous congestion
12) Atelectasis
13) Post traumatic ribs deformation
14) Atherosclerosis of the aorta
15) Pneumosclerosis
16) Abscess
17) Cardiomegaly
18) Sarcoidosis

TABLE I: Dataset Splitting

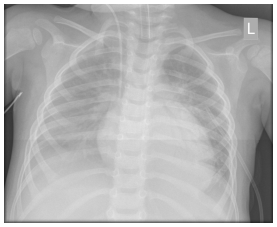
X-Ray Chest Dataset		
Dataset	**Percentage**	**N° of Images**
Train	60%	7092
Validation	20%	2365
Test	20%	2365
Total	100%	11822

B. Data Cleaning

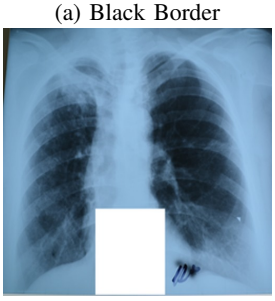
In an effort to mitigate the risk of misclassification during the training phase of our project, a critical initial step involved meticulously cleansing the dataset of extraneous features deemed non-essential for our analysis. This process required particular attention to three specific elements commonly present in the images: the elimination of black borders, the removal of any white boxes, and the careful extraction of white borders typically surrounding X-ray images. These measures were essential to improve the accuracy and effectiveness of the model's training process.



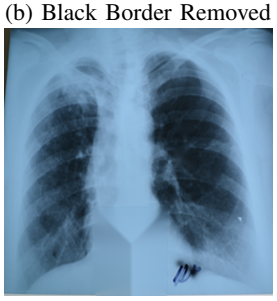
(a) Black Border



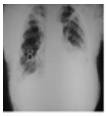
(b) Black Border Removed



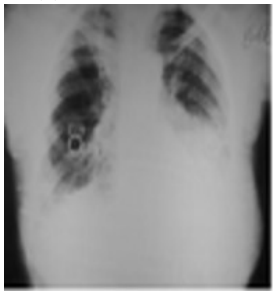
(c) White Box



(d) White Box Filled



(e) White Border



(f) White Border Removed

Fig. 3: (a), (b), (e) and (f) are illustrating the elimination of black and white borders. (c) and (d) are showing how a white box its being filled.

Initially the process was started by manually identifying images containing borders and white boxes. After this stage, the images with borders were run through a detection and precisely cropped to remove these extraneous elements. For images with white boxes, a specialized technique was implemented, leveraging the features surrounding the boxes to fill in the resultant empty spaces. This operation marked the initial phase of our comprehensive data cleaning process. In the subsequent stage, all images were converted to the RGB color space and uniformly resized to the dimensions of 224x224 pixels. This standardization was pivotal for TensorFlow compatibility and also was executed while meticulously preserving the original aspect ratio of each image. The final step in our preparatory phase involved a thorough normalization of pixel values across the dataset, a crucial measure to ensure consistency in the input data for our forthcoming analytical procedures.

C. Data Augmentation

In order to address the imbalance of the data and to foster the generalizability of the results post-training, the implementation of data augmentation techniques is integral in the pre-processing phase of machine learning tasks. These augmentation methods, designed to introduce variability and broaden the training dataset's diversity, encompass a range of transformations, including but not limited to:

- Rotation
- Zoom Range
- Horizontal Flip
- Contrast Adjustment
- Elastic Deformation

In this particular project, a strategic application of these data augmentation methods was undertaken during the pre-training phase, specifically targeting only the class with a comparatively lower number of images to balance the dataset. This systematic approach to data augmentation plays a fundamental role in enhancing model robustness and ensuring a more robust and reliable analytical outcome.

TABLE II: Data Augmentation Parameters

Parameters	**Value**
Rotation	15
Zoom Range	0.2
Horizontal Flip	True
Contrast Adjustment	[0.8, 1.2]
Width Shift	0.1
Height Shift	0.1
Shear	5

D. Undersampling and Oversampling

Class imbalance in datasets is a prevalent issue in machine learning, often resulting in models that are skewed towards majority classes. Such imbalance can severely impact the model's ability to generalize, leading to suboptimal performance. Addressing this imbalance is crucial for developing robust and fair models. Our proposed 'balance classes' function addresses

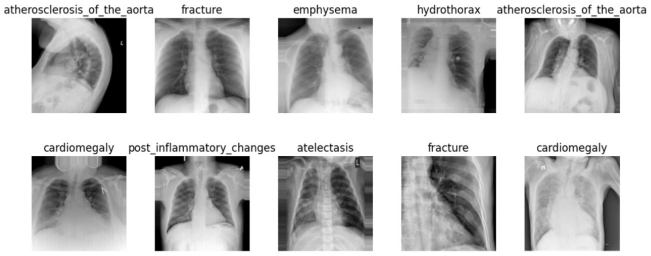


Fig. 4: Sample Augmented Images

this challenge by employing a dual strategy of oversampling minority classes and undersampling majority classes.

*a) **Oversampling Small Classes**:* : For classes with occurrences below the 'small class threshold', the function amplifies their presence by replicating existing data points. The replication factor, or multiplier, is dynamically calculated based on the class count, ensuring that these classes reach a numerical strength closer to the defined threshold.

*b) **Undersampling Large Classes**:* : Conversely, classes with counts exceeding the 'large class threshold' undergo a mild reduction in their representation. This is executed probabilistically, where each instance has a set chance (currently 0.7) of being included in the balanced dataset, thereby subtly diminishing the predominance of these classes.

*c) **Exemption for Middle-Sized Classes**:* : Classes falling within the thresholds are deemed adequately represented and are thus left unaltered.

In conclusion, while the 'balance classes' function marks a significant stride towards rectifying class imbalances within datasets, it is important to acknowledge that achieving a perfectly balanced dataset remains a challenging endeavor. This residual imbalance, although reduced, still presents a nuanced challenge for machine learning models. In this project, the 'balance classes' function serves as a valuable tool, contributing significantly to our capacity to handle class imbalances more effectively, even if absolute balance remains an elusive goal.

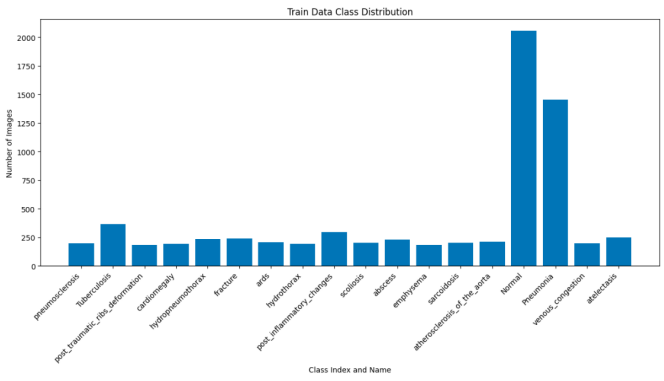


Fig. 5: Class Distribution for Train Data

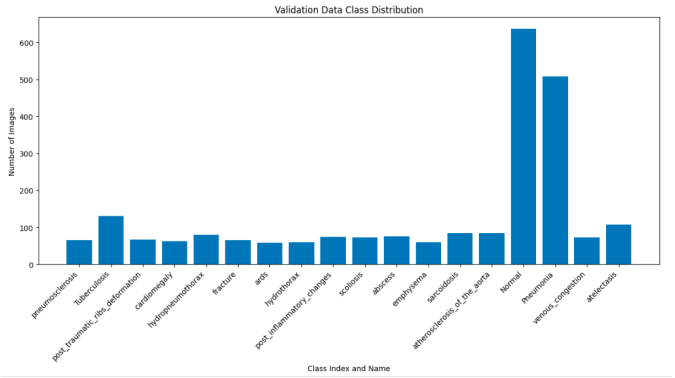


Fig. 6: Validation Data

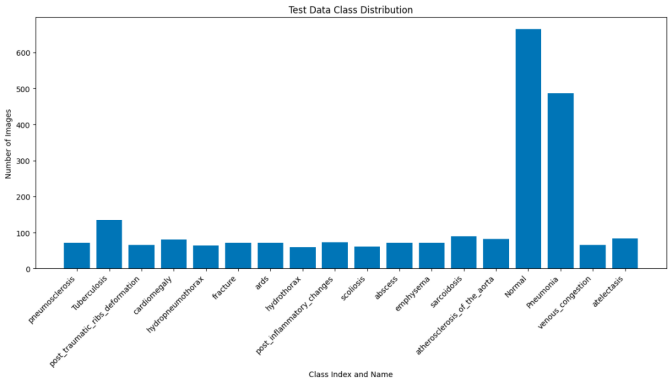


Fig. 7: Test Data

E. Test Bench

The entire training and testing process is done in Google Colab GPU. TensorFlow and Keras API were used for implementation. For GPU acceleration, nvidia-smi library is used in order to speed up the training process. Two different training runs are made for both ResNet-50 and DenseNet-121 pre-trained models. The training time for ResNet-50 and DenseNet-121 were approximately 20/30 minutes respectively.

F. Models Implementation

The ResNet-50 model is utilized with pre-loaded ImageNet weights. The top layers of the network are set to be trainable, allowing the model to fine-tune on our specific dataset. This model is augmented with additional dense layers, batch normalization, ReLU activation functions, and dropout layers for regularization. Similarly, we employ the DenseNet121 architecture with ImageNet weights. In this case, we freeze all the base layers during training to keep the learned features intact and append our own dense layer with a ReLU activation followed by a dropout layer. Both models are compiled with the Adam optimizer and categorical crossentropy loss function, suitable for multi-class classification tasks. We also track several performance metrics, including:

1) accuracy
2) precision
3) recall

4) F1 score

G. Hypertuning

To rigorously assess the performance of this pre-trained models, we conducted a series of experiments. The initial experiment focused on evaluating the impact of varying the number of training epochs (10 and 20) on the models' performance. Subsequently, we conducted an analysis to determine the effect of different batch sizes (32 and 64) on the models' training efficacy.

Model	Epoch	Precision	Recall	F1-score	Accuracy
ResNet50	10	0.81	0.37	0.40	0.29
DenseNet121	10	0.98	0.99	0.98	0.96
ResNet50	20	0.83	0.78	0.76	0.78
DenseNet121	20	0.98	0.99	0.99	0.97

TABLE III: Comparison of ResNet50 and DenseNet121 with different training configurations and batch size 32.

Model	Epoch	Precision	Recall	F1-Score	Accuracy
ResNet50	10	0.99	0.99	0.99	0.97
DenseNet121	10	0.98	0.99	0.98	0.96
ResNet50	20	0.26	0.10	0.06	0.16
DenseNet121	20	0.99	0.99	0.99	0.97

TABLE IV: Comparison of ResNet50 and DenseNet121 with different training configurations and batch size 64.

H. Evaluation and Results

In this section, we delve into a comprehensive evaluation of the DenseNet-121 and ResNet-50 models, focusing on their learning curve smoothness, indicators of overfitting, and overall performance as evidenced by classification metrics. Our analysis is grounded in the comparison of these models under the same operational conditions, providing insights into their relative efficiencies and robustness in handling complex image classification tasks. We first observed the progression of loss and accuracy across training epochs for DenseNet-121.

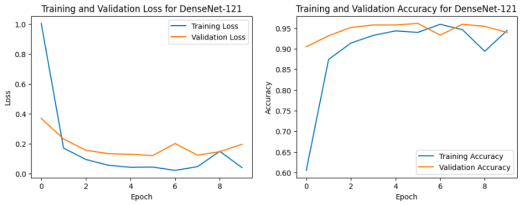


Fig. 8: DenseNet-121 Loss and Accuracy during Train and Validation with batch size 32 and epoch 10 (best representation of DenseNet-121 behavior during training).

The model exhibited a consistent and smooth learning curve, devoid of significant spikes or erratic fluctuations. Such smoothness in learning curves is indicative of a stable and effective learning process, suggesting that the model's learning parameters, including the learning rate and batch size, are optimally tuned to ensure a steady and gradual improvement.

In contrast, the ResNet-50 models exhibited notable fluctuations in validation loss, raising concerns about potential overfitting or instability in its learning process.

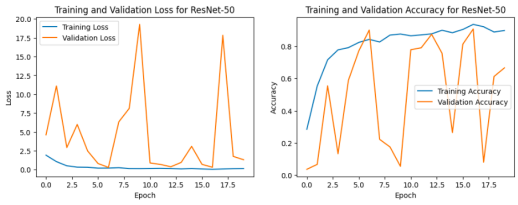


Fig. 9: ResNet-50 Loss and Accuracy during Train and Validation with batch size 32 and epoch 20 (best representation of ResNet-50 tendency to overfit during training).

The DenseNet-121 model stands out for its stable training behavior, evidenced by consistent improvements in loss and accuracy. Its performance, marked by the absence of erratic behavior in validation metrics, indicates a strong generalization capability, making it a reliable option for deployment in scenarios requiring robustness against new and unseen data. Conversely, the ResNet-50 model, as shown in the subsequent graphs and tables, demonstrates a different pattern in its accuracy and loss metrics, warranting a detailed examination.

Condition	Precision	Recall	F1-score	Support
Normal	0.98	0.64	0.78	721
Pneumonia	0.75	0.95	0.84	478
Tuberculosis	0.64	0.95	0.76	131
Abscess	0.67	0.86	0.76	74
ARDS	1.00	0.81	0.89	78
Atelectasis	0.90	0.45	0.60	84
Atherosclerosis of the Aorta	0.86	0.95	0.90	74
Cardiomegaly	1.00	0.26	0.41	70
Emphysema	1.00	0.78	0.88	54
Fracture	0.74	0.96	0.84	84
Hydropneumothorax	1.00	0.71	0.83	69
Hydrothorax	0.31	1.00	0.48	66
Pneumosclerosis	0.81	0.94	0.87	62
Post Inflammatory Changes	0.95	0.52	0.67	81
Post Traumatic Ribs Deformation	0.91	0.83	0.87	59
Sarcoidosis	0.80	0.99	0.89	75
Scoliosis	0.65	0.98	0.78	66
Venous Congestion	1.00	0.52	0.68	58
Accuracy			0.78	2384
Macro Avg	0.83	0.78	0.76	2384
Weighted Avg	0.85	0.78	0.78	2384

TABLE V: Classification Metrics for ResNet-50, batch size 32 and epoch 20.

The results presented in the DenseNet-121 table corroborate the superior performance of this CNN architecture when compared to that of ResNet-50. The data clearly demonstrates the enhanced efficacy of DenseNet-121 in key performance metrics.

Condition	Precision	Recall	F1-score	Support
Normal	0.96	0.93	0.94	721
Pneumonia	0.96	0.94	0.95	478
Tuberculosis	0.80	0.98	0.88	131
Abscess	0.99	1.00	0.99	74
ARDS	1.00	1.00	1.00	78
Atelectasis	1.00	1.00	1.00	84
Atherosclerosis of the Aorta	1.00	1.00	1.00	74
Cardiomegaly	0.99	1.00	0.99	70
Emphysema	1.00	1.00	1.00	54
Fracture	0.99	1.00	0.99	84
Hydropneumothorax	1.00	1.00	1.00	69
Hydrothorax	0.99	1.00	0.99	66
Pneumosclerosis	0.97	1.00	0.98	62
Post Inflammatory Changes	0.99	0.99	0.99	81
Post Traumatic Ribs Deformation	1.00	1.00	1.00	59
Sarcoidosis	1.00	1.00	1.00	75
Scoliosis	1.00	1.00	1.00	66
Venous Congestion	0.95	1.00	0.97	58
Accuracy			0.96	2384
Macro Avg	0.98	0.99	0.98	2384
Weighted Avg	0.97	0.96	0.98	2384

TABLE VI: Classification Metrics for DenseNet-121, batch size 32 and epoch 10.

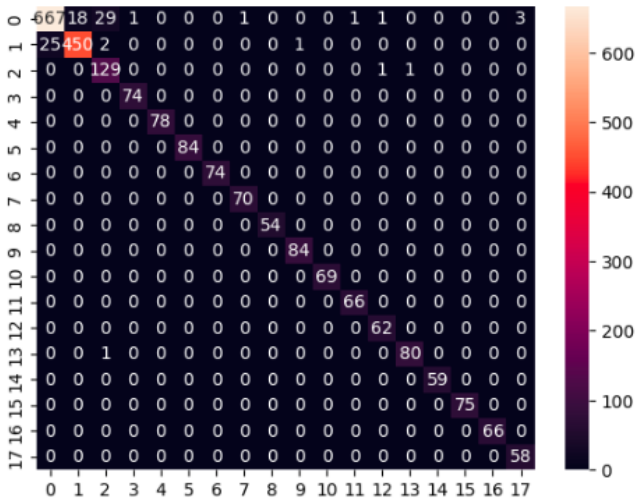


Fig. 10: DenseNet-121 Confusion Matrix

IV. Conclusion

In this study, we embarked on a comparative analysis of two prominent Convolutional Neural Network (CNN) architectures, ResNet and DenseNet, within the realm of chest X-ray image classification. This comparison was not merely an academic exercise but a practical evaluation aimed at discerning the most effective model for medical image analysis, a critical component in modern diagnostic procedures.

Through meticulous experimentation and analysis, our project illuminated the distinct characteristics and performance metrics of both architectures. The DenseNet-121 model, known for its densely connected layers, demonstrated exceptional stability and consistency in its training process. The learning curves of this model were notably smooth, indicating an effective capture of features and patterns without the hindrance of overfitting. This was further substantiated by the close alignment of training and validation losses, a testament to its generalization capabilities.

On the other hand, the ResNet-50 model, while powerful in its own right, exhibited certain limitations in this specific application. Notable fluctuations in the validation loss pointed towards potential overfitting issues or instability in learning, which could impede its effectiveness in a real-world medical diagnostic context.

The findings of this project are significant for several reasons. Firstly, they contribute valuable insights into the ongoing development of AI in medical diagnostics, particularly in the efficient and accurate classification of diseases from chest X-ray images. Secondly, the study underscores the importance of choosing the right model architecture that aligns with the specific nuances and demands of medical image analysis.

References

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

[2] "Chest X-ray – 17 Diseases" Kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/trainingdatapro/chest-xray-17-diseases

[3] "Chest X-Ray Images (Pneumonia)", Kaggle.com. [Online]. Available: https://www.kaggle.com/datasets/paultimothymooney/chest-xray-pneumonia

[4] "Tuberculosis (TB) Chest X-ray Database", Kaggle.com.[Online]. Available: https://www.kaggle.com/datasets/tawsifurrahman/tuberculosis-tb-chest-xray-dataset

[5] NIAID TB portal program dataset [Online]. Available:https://tbportals.niaid.nih.gov/download-data

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