الجمهورية الجزائرية الديمقراطية الشعبية

PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA

وزارة التعليم العالي و البحث العلمي

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

جامعة عمّار ثليجي بالأغواط

UNIVERSITY OF AMAR TELIDJI LAGHOUAT

كلية العلوم

FACULTY OF SCIENCES

قسم الإعلام الآلي

DEPARTMENT OF COMPUTER SCIENCES

Master Thesis

Domain Mathematics and Computer Science

Field Computer Science

Option Decision-making Information Systems

By:

Zaid Ahmed Amine

Topic

Enhancing Chest Disease Detection through the Synergy of Deep Learning and Genetic Algorithms

---

Defended Publicly in Front of the Jury Composed of

M_r.	Y. Ouinten	Prof.	PRESIDENT	UATL
M_r.	M. BOUZIDI	MCB	REVIEWER	UATL
M_r.	L. BENSAAD	MCB	REVIEWER	UATL
M_{rs}.	H. CHERROUN	Prof.	SUPERVISOR	UATL

Academic Year 2022/2023

Dedications

I dedicate this modest work to the two people who have never stopped sacrificing themselves for me to succeed.

To my dear mother and my father, whom no dedication can express what I owe them, for their benevolence, their affection, and their support of the treasures of benevolence, generosity, which supported me during all my years of studies, may God keep them and bless them..

To my dear sisters, all the members of my family, and to all my dear friends who have helped me during all my studies.

And to all who have taught me throughout my educational life, And to my second-year master's classmates in general.

Thank you all.

Zaid Ahmed Amine .

Acknowledgements

With utmost humility and gratitude, we commence this undertaking in the name of Allah, the Most Gracious, the Most Merciful. We acknowledge the boundless blessings and guidance bestowed upon us by our Creator, and we humbly recognize that our accomplishments are solely a result of His infinite mercy and favor. We beseech His continued blessings and guidance as we embark on this journey and in all our future endeavors.

We would also like to express our sincere appreciation to our supervisor, whose invaluable advice and guidance have been instrumental throughout the entirety of this undertaking..

Our heartfelt thanks are extended to all individuals who have directly or indirectly contributed to the successful completion of this work.

Lastly, we wish to convey our deep appreciation to our families, whose unwavering support has been a constant source of motivation, as well as to every individual who has played a role in the realization of this dissertation. Additionally, we extend our gratitude to all the educators who have contributed to our intellectual development and growth.

Thank you all.

Zaid Ahmed Amine .

Abstract

Chest X-ray (CXR) imaging plays a pivotal role in modern healthcare, serving as a prominent and widely employed diagnostic modality. Its application is instrumental in aiding radiologists to discern various critical pulmonary conditions, enabling visual scrutiny of CXR images to identify disease manifestation. However, the intricate nature of lung diseases, characterized by resemblant patterns and symptoms, poses significant diagnostic challenges that may potentially result in severe ramifications if misinterpreted. Deep learning (DL) models have surfaced as auspicious prediction methodologies, exhibiting remarkable precision akin to human-level performance. In this work, we are addressing the problem of lung disease detection using CXR images by employing a custom Deep Convolutional Neural Network (DCNN) architecture, we employ the model to extract intricate feature representations and accurately classify potential diseases within the CXR images. We are also deploying one of the Python Genetic Algorithm (GA) library PyGAD modules which is the pygad.gacnn module used for training DCNNs using the GA. The targeted dataset is the COVID-19-Radiography database. We investigate the deployment of the GA approach on three different dataset samples. The results demonstrated the effect of the utilization of the GA allows us to achieve optimal outcomes by identifying the optimal set of hyperparameters and selecting the fittest individuals from each generation. Alternatively, the integration of specialized library modules, similar to the one employed in our investigation, enables the construction and training of CNN using the GA, facilitating the attainment of accurate results. The evaluation showed that the classification accuracy of our genetic algorithm approach achieved an Accuracy (ACC) of 77.31%, 78.23%, and 78.87%, improving the performances by 0.51%, 1.56%, and 1.7% for the three proposed samples respectively.

Keywords: Chest Disease, Chest-Xray, Deep Learning, Genetic Algorithm, Convolutional Neural Network.

مُلخّص

يعتبر تصوير الصدر بالأشعة السينية أحد أكثر طرق التصوير الطبي استخدامًا. يساعد بشكل كبير أخصائيي الأشعة في الكشف عن العديد من أمراض الرئة التي تهدد الحياة ويساعدهم على الفحص البصري للصور. يلعب تصوير الصدر بالأشعة السينية دورًا محوريًا في الرعاية الصحية الحديثة ، حيث يعمل كوسيلة تشخيصية بارزة ومستخدمة على نطاق واسع. يعد تطبيقه مفيدًا في مساعدة أخصائيي الأشعة على تمييز مختلف الحالات الرئوية الحرجة ، مما يتيح الفحص البصري لصور الصدر بالأشعة السينية لتحديد مظاهر المرض. ومع ذلك ، فإن الطبيعة المعقدة لأمراض الرئة ، التي تتميز بأنماط وأعراض متشابهة ، تطرح تحديات تشخيصية كبيرة قد تؤدي إلى تداعيات شديدة إذا أسيء تفسيرها. ظهرت نماذج التعلم العميق كمنهجيات تنبؤ ميمونة ، وأظهرت دقة ملحوظة مماثلة للأداء على مستوى الإنسان. في هذا العمل ، نعالج مشكلة اكتشاف أمراض الرئة باستخدام صور الصدر بالأشعة السينية من خلال استخدام بنية شبكة عصبية تلافيفية عميقة مخصصة ، ونستخدم النموذج لاستخراج تمثيلات الميزات المعقدة وتصنيف الأمراض المحتملة بدقة في صور الأشعة السينية للصدر. . نقوم أيضًا بنشر إحدى وحدات بايغاد الخاصة بمكتبة الخوارزميات الجينية في بايثون وهي وحدة بايغاد الخوارزمية الجينية للشبكات العصبية الالتفافية المستخدمة لتدريب الشبكات العصبية التلافيفية العميقة باستخدام الخوارزمية الجينية. مجموعة البيانات المستهدفة هي قاعدة بيانات كوفيد 19 للتصوير الشعاعي. نحن نحقق في نشر نهج الخوارزمية الجينية على ثلاث عينات مختلفة من مجموعات البيانات. أظهرت النتائج أن تأثير استخدام الخوارزمية الجينية يسمح لنا بتحقيق النتائج المثلى من خلال تحديد المجموعة المثلى للمعلمات الفائقة واختيار أفضل الأفراد من كل جيل. بدلاً من ذلك ، فإن تكامل وحدات المكتبة المتخصصة ، على غرار تلك المستخدمة في تحقيقنا ، يتيح بناء وتدريب الشبكة العصبية التلافيفية باستخدام الخوارزمية الجينية ، مما يسهل تحقيق نتائج دقيقة. أظهر التقييم أن دقة تصنيف نهج الخوارزمية الجينية لدينا حققت دقة 77.31% و 78.23% و 78.87% ، مما أدى إلى تحسين الأداء بنسبة 0.51% و 1.56% و 1.7% للعينات الثلاثة المقترحة. على التوالى.

الكلمات المفتاحيّة:

أمراض الصدر ، تصوير الصدر بالأشعة السينية ، التعلم العميق ، الخوارزمية الجينية ، الشبكة العصبية التلافيفية.

Résumé

La radiographie pulmonaire (CXR) joue un rôle central dans les soins de santé modernes, servant de modalité de diagnostic importante et largement utilisée. Son application est essentielle pour aider les radiologues à discerner diverses conditions pulmonaires critiques, permettant un examen visuel des images CXR pour identifier la manifestation de la maladie. Cependant, la nature complexe des maladies pulmonaires, caractérisées par des schémas et des symptômes similaires, pose des défis diagnostiques importants qui peuvent potentiellement entraîner de graves ramifications si elles sont mal interprétées. Les modèles d'apprentissage en profondeur (DL) sont apparus comme des méthodologies de prédiction de bon augure, présentant une précision remarquable proche des performances au niveau humain. Dans ce travail, nous abordons le problème de la détection des maladies pulmonaires à l'aide d'images CXR en utilisant une architecture personnalisée de réseau de neurones à convolution profonde (DCNN), nous utilisons le modèle pour extraire des représentations de caractéristiques complexes et classer avec précision les maladies potentielles dans les images CXR. Nous déployons également l'un des modules PyGAD de la bibliothèque Python Genetic Algorithm (GA), qui est le module pygad.gacnn utilisé pour former les DCNN à l'aide de l'AG. Le jeu de données ciblé est la base de données COVID-19-Radiographie. Nous étudions le déploiement de l'approche GA sur trois échantillons de jeux de données différents. Les résultats ont démontré que l'effet de l'utilisation de l'AG nous permet d'obtenir des résultats optimaux en identifiant l'ensemble optimal d'hyperparamètres et en sélectionnant les individus les plus aptes de chaque génération. Alternativement, l'intégration de modules de bibliothèque spécialisés, similaires à celui utilisé dans notre enquête, permet la construction et la formation de CNN à l'aide de l'AG, facilitant l'obtention de résultats précis. L'évaluation a montré que la précision de classification de notre approche d'algorithme génétique a atteint une Accuracy (ACC) de 77,31%, 78,23% et 78,87%, améliorant les performances de 0,51%, 1,56% et 1,7% pour le respectivement trois échantillons proposés.

Mots-clés: Maladie thoracique, radiographie thoracique, apprentissage en profondeur, algorithme génétique, réseau de neurones convolutifs..

Table of Contents

List of Figures

List of Tables

List of Equations

Abbreviations

AI	Artificial intelligence
ML	Machine Learning
DL	Deep Learning
CV	Computer Vision
NLP	Natural Language Processing
RL	Reinforcement Learning
CNN	Convolutional Neural Network
ANN	Artificial Neural Network
ACC	Accuracy
AUC	Area Under Curve
MAE	Mean Absolute Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error
HITL	Human in the Loop
GA	Genetic Algorithm
CXR	Chest X-ray Radiography
JSRT	Japanese Society of Radiological Technology
STR	Society of Thoracic Radiology
GAN	Generative Adversarial Networks
TB	Tuberculosis
DCNN	Deep Convolutional Neural Network
RSNA	Radiological Society of North America
HRCT	High-Resolution Computed Tomography
DC-GAN	Deep Convolution Adversarial Networks
CLAHE	Adaptive Contrast-Limited Histogram Equalization
BCET	Balance Contrast Enhancement Technique
WHO	World Health Organization

Introduction

In the realm of medical diagnostics, Chest X-Ray (CXR) imaging emerges as a rapid and economical modality frequently employed by radiologists for the examination and evaluation of various anatomical structures encompassing the cardiovascular system, respiratory organs, skeletal framework, vascular network, and pulmonary conduits [1]. It plays a critical role in greatly assisting radiologists in detecting several life-threatening lung diseases.

In recent times, deep learning (DL) techniques have been extensively utilized in numerous research investigations to scrutinize and interpret Chest X-Ray (CXR) scans, particularly in the domain of image categorization. A considerable body of prior scholarly works has concentrated on training DL models using meticulously devised convolutional neural network (CNN) structures, notably including VGG [2], GoogleNet [3], ResNet [4], and DenseNet [5] [6]. The effectiveness of a deep learning model can be significantly impacted by various factors encompassing the depth of the convolutional neural network (CNN) architecture, techniques employed for data augmentation, approaches utilized for input processing, dimensions of the images, and strategies implemented for model training. By using methods that can help improve model performance, several investigations have attained innovative methodologies that showcase distinctiveness in their approaches [6]. For example, Wang et al. [7] CNN models were trained to utilize the whale optimization algorithm. This technique effectively addresses challenges associated with extensive manual parameter tuning and parallelization of training procedures encountered in conventional gradient descent-based methods [6]. Khishe et al. [8] introduced a highly efficient biogeography-inspired optimization technique to automatically fine-tune hyperparameters in algorithms. This approach addresses the labor-intensive task of manually selecting parameters such as the number of output channels, kernel size, layer type, training convolution rate, and batch size, resulting in improved optimization processes. Despite the capability of chest X-ray (CXR) imaging to identify various thoracic diseases, developing methods specifically designed to classify multiple disease indicators still needs to be improved [6].

The resemblance and overlapping symptoms among numerous chest diseases pose challenges to accurate diagnosis, potentially resulting in critical errors and adverse outcomes. Deep learning models have shown great promise as predictive tools, exhibiting a level of precision comparable to human expertise. Nonetheless, the progress of deep

learning (DL) methodologies for image analysis is impeded by a multitude of intrinsic constraints and limitations [6]. A significant drawback revolves around the scarcity of extensive chest X-ray (CXR) datasets, posing a prominent limitation. While deep learning models necessitate a substantial number of parameters, the limited availability of CXR imaging data increases the risk of model overfitting [6]. As well as the usual challenges such as classification in complex images, symptoms segmentation, labeled images challenges can be difficult and expensive, and the exploitation of multiple information sources.

The primary objective of this research is to implement a multi-class classification framework on a meticulously balanced collection of CXR images obtained from the COVID-19-Radiography database [9, 10], To achieve this, we thoroughly examine a tailored DCNN architecture capable of extracting distinctive feature representations and effectively classifying various diseases depicted in these CXR images. Furthermore, we employ a specific module, "pygad.gacnn", from the Python Genetic Algorithm (GA) Library, known as PyGAD [11], to facilitate the training of our DCNN models using the GA optimization techniques. These techniques greatly assist radiologists in detecting several life-threatening lung diseases and help them visually inspect CXR images to detect the presence of disease. As well as helping to build and train a compact and efficient CNN model using the genetic algorithm.

Previous studies have generally used optimization techniques either to train CNN models or to focus more on parameter tuning. In this study, we develop an approach to train CNNs using GA and parallelize the process of refining and fine-tuning hyperparameters at the same time.

In addition to this introduction, this manuscript is organized into four chapters arranged as follows:

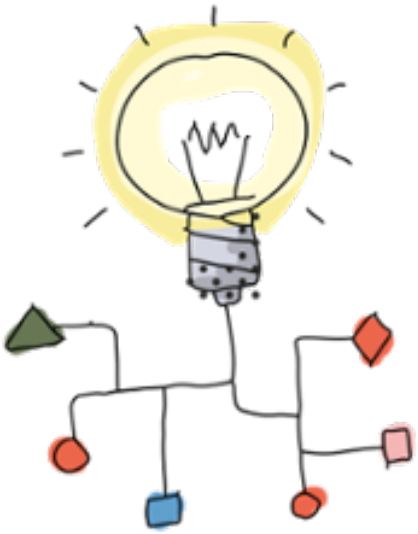
- The first chapter of this manuscript is dedicated to recalling some preliminaries on Deep Learning. Presenting a detailed description of some basics of Convolutional Neural Networks (CNNs) and their related metrics, Then giving an overview of genetic algorithms, and finally presenting types of classification problems in machine learning related to our application.
- In the second chapter, we start by stating the most common and publicly available CXR datasets, subsequently, we present a comprehensive overview of prevalent approaches employed for preprocessing CXR images, encompassing enhancement techniques and data augmentation strategies. These methods aim to enhance the quality of images and augment the available data, as well as an overview of state-of-the-art deep learning models that leverage CXR images for the detection and diagnosis of various chest diseases. And finally, a comprehensive discussion where we discuss and interpret explainability is provided.
- In the final chapter, we present the details of our proposed approach, starting with

the selection of datasets from the pre-existing publicly available datasets, the data preparation and pre-processing phase, then diving into our proposed strategy. This involves two steps: the creation of the proposed DCNN architecture as well as the application of a genetic algorithm to refine and tune the hyper-parameters of the targeted problem. The various performance metrics used to evaluate the results are described, as well as the tools and environments used for development. Finally, we present the results of the project carried out in our study. When we talk about the applied test set, highlight the effects of applying a genetic algorithm to our model and conclude with a thorough discussion of all of the above.

- In the conclusion, we summarize the main results and point out some perspectives.

Chapter 1

Generalities



In this chapter, we recall some preliminaries on Deep Learning. A detailed description of some basics of Convolutional Neural Networks (CNNs) and their related metrics are also presented, Then we give an overview of genetic algorithms and finally present types of classification problems in machine learning related to our application.

1 Deep Learning

In the current landscape of the business realm, enterprises leverage their unique capabilities and inventive ideas to construct intelligent machines and applications. Artificial intelligence (AI), Machine Learning (ML), and Deep Learning (DL) have gained significant prominence in contemporary business discussions on a global scale. These three technological concepts are frequently subject to interchangeable usage. However, it is crucial to discern that they do not precisely encompass identical domains.
Figure 1.1 illustrates the main differences between AI, ML, and DL.

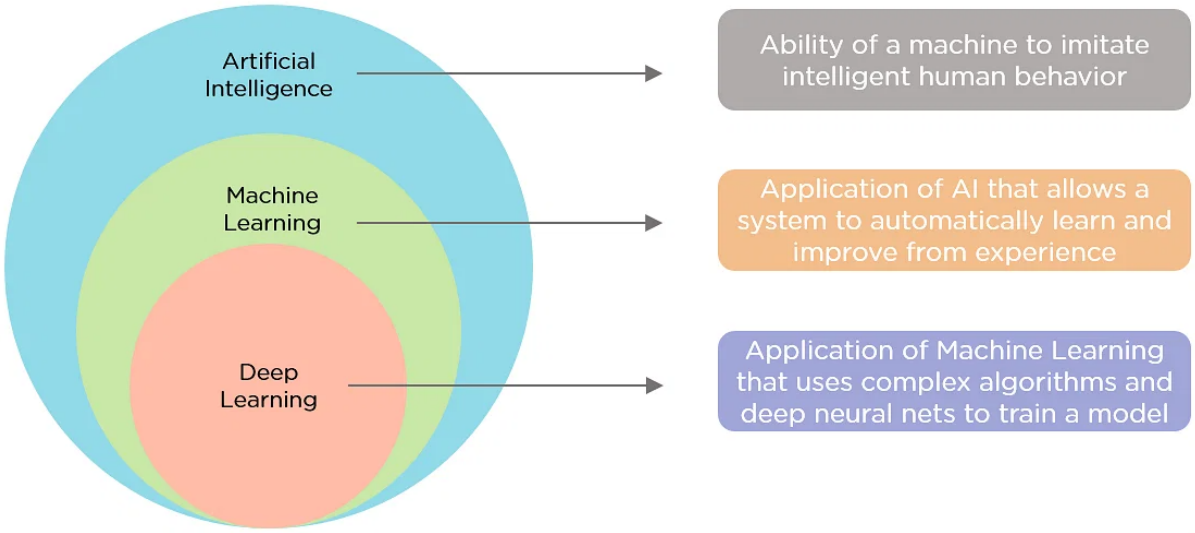


Figure 1.1: The Difference Between AI, ML, and DL
[12]

AI revolves around the notion of constructing intelligent machines capable of addressing common challenges. ML, on the other hand, constitutes a subset of AI that facilitates the development of applications driven by artificial intelligence. Finally, DL represents a branch of ML that relies on artificial neural networks, utilizing extensive datasets and intricate algorithms to train models.

In our research, we primarily focus on the domain of **Deep Learning (DL)**, which encompasses applications in supervised, unsupervised, and reinforcement machine learning. Within this realm, various techniques such as neural networks and their diverse forms are employed to address the challenges at hand.

- **Main Applications of DL** can be categorized into distinct domains, namely Computer Vision (CV), Natural Language Processing (NLP), and Reinforcement Learning (RL).
  1. Computer Vision: Where we can see Object detection and recognition, Image classification, And Image segmentation.
  2. Natural Language Processing: in NLP, DL exhibits various prominent applications, including but not limited to automatic text generation, speech recognition, and sentiment analysis.
  3. Reinforcement learning: where we'll be more focused on Robotics, Game playing, and Control systems.
- **Limitations of DL** Even with the significant advancements of DL in various fields we still face some setbacks and challenges. Some of the main challenges are:
  - Data availability: where DL necessitates substantial quantities of data to achieve optimal performance. Additionally, more advanced and precise models often demand an increased number of parameters, posing significant challenges in numerous domains and scenarios.
  - Time-consuming: which differentiates from one problem to another.
  - Computational Resources Cost: For training the deep learning model.
  - Overfitting: upon repeated training iterations, the model's adaptability diminishes, rendering it incapable of handling multitasking scenarios. While it may yield efficient and precise outcomes, it becomes excessively specialized and limited to addressing a singular problem, primarily due to its training data-centric nature.

2 Convolutional Neural Network (CNN)

CNN is one of the most common and efficient DL techniques for processing image-based models.

2.1 Backgrounds of CNN

CNNs were initially conceived and utilized in the 1980s, primarily excelling in the task of handwritten digit recognition. The postal sector predominantly employed CNNs for reading zip codes, pin codes, and related applications. Notably, it is crucial to emphasize that deep learning models, including CNNs, necessitate substantial volumes of training data and significant computational resources. This particular requirement posed a significant limitation for CNNs during that era, consequently restricting their widespread adoption beyond the confines of the postal sector and impeding their integration into the broader domain of machine learning [13].
In 2012, Alex Krizhevsky recognized the opportune moment to revive the branch of deep learning that leverages multi-layered neural networks. The resurgence was

propelled by the availability of vast datasets, such as the ImageNet collection containing millions of meticulously labeled images, and the abundance of computing resources at researchers' disposal. These influential elements played a crucial role in rejuvenating and reinvigorating the utilization of CNNs. [13].

2.2 Functional Definition of CNN

CNNs share a fundamental resemblance to conventional Artificial Neural Networks (ANNs), as they consist of neurons that undergo self-optimization through the process of learning [14]. A distinguishing characteristic between CNNs and traditional ANNs lies in their predominant application within the domain of image pattern recognition. This specialization empowers CNNs to encode image-specific features into their architecture, rendering them more adept at tasks centered around images. Consequently, this specialized focus enables a notable reduction in the parameter count necessary for model configuration, further streamlining the process [14].

2.3 The Benefits of Utilizing CNN

CNNs prove highly valuable in scenarios where extensive datasets necessitate the application of image recognition, image classification, or computer vision (CV) techniques. Furthermore, they demonstrate notable effectiveness in classifying audio, temporal, and signal data with remarkable proficiency [13].

2.4 CNN Architecture

A CNN architecture is comprised of layers such as convolutional layers, pooling layers, and fully-connected layers.
A simple CNN architecture for the MNIST database [1] which is a classification of a large collection of handwritten digits is illustrated in Figure 1.2.

[1] Modified National Institute of Standards and Technology database

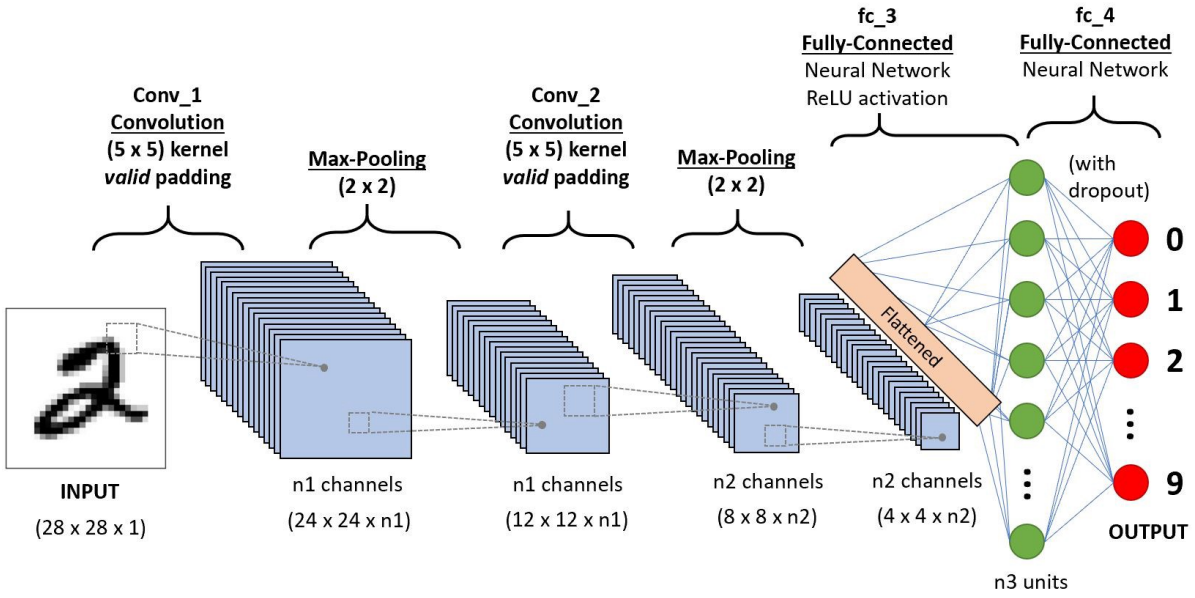


Figure 1.2: A Simple CNN Architecture Example
[15]

The fundamental functionality of the aforementioned CNN example can be dissected into four primary components:

- The initial layer, known as the input layer, serves as a container for storing the pixel values associated with the image.
- **The Convolutional Layer:** serving as the fundamental building block of a CNN, assumes a pivotal role by hosting the bulk of computational operations. In the Convolutional layer, the input images undergo a series of convolutions with dedicated filters, resulting in the activation of distinct features embedded within the images [14].
  The rectified linear unit (often abbreviated as ReLU) is employed to apply an activation function, such as the sigmoid function, to the output generated by the preceding layer's activation [14].
  A simple Convolutional layer example is illustrated in Figure 1.3.

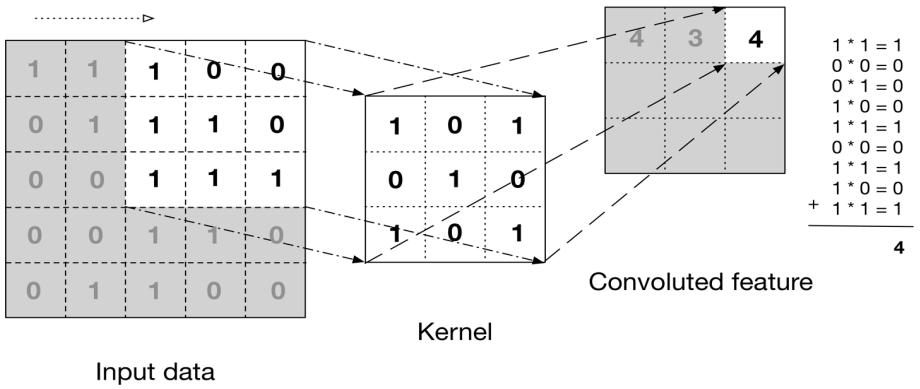


Figure 1.3: A Simple Convolutional Layer Example
[13]

- **The Pooling Layer :** similar to the convolutional layer, assumes the role of reducing the spatial dimensions of the convolved feature through downsampling, thereby diminishing the computational burden associated with data processing. This downsampling operation effectively simplifies the output, resulting in a reduction of the network's parameter complexity, thus streamlining the learning process. Consequently, the pooling layer efficiently streamlines the network architecture.
  Within the context of pooling, two distinct types are commonly employed: average pooling and max pooling.
  A simple pooling layer example is illustrated in Figure 1.4.

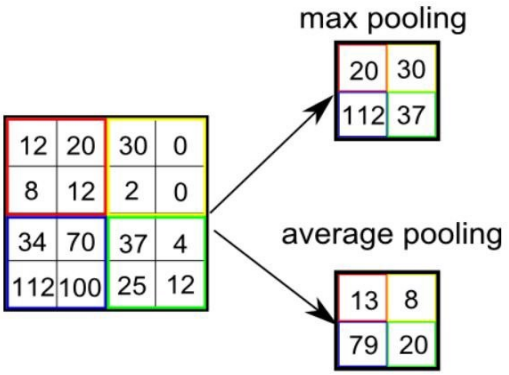


Figure 1.4: A Simple Pooling Layer Example
[13]

- **The Fully Connected Layer :** It's where the CNN attempt to produce class scores from the activation just like a standard ANNs for example image classification. The term "Fully Connected" denotes a configuration where each input or node from a given layer is intricately linked to every activation unit or node within the subsequent layer.
  An example of a fully connected layer as well as an example of a simple CNN architecture that exhibits the two phases of future training and classification are

shown in Figure 1.5 and Figure 1.6 respectively.

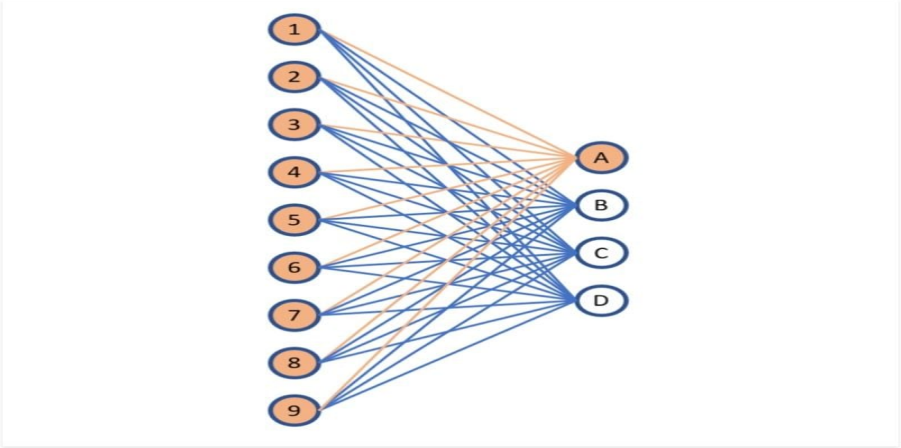


Figure 1.5: A Simple Fully Connected Layer Example
[16]

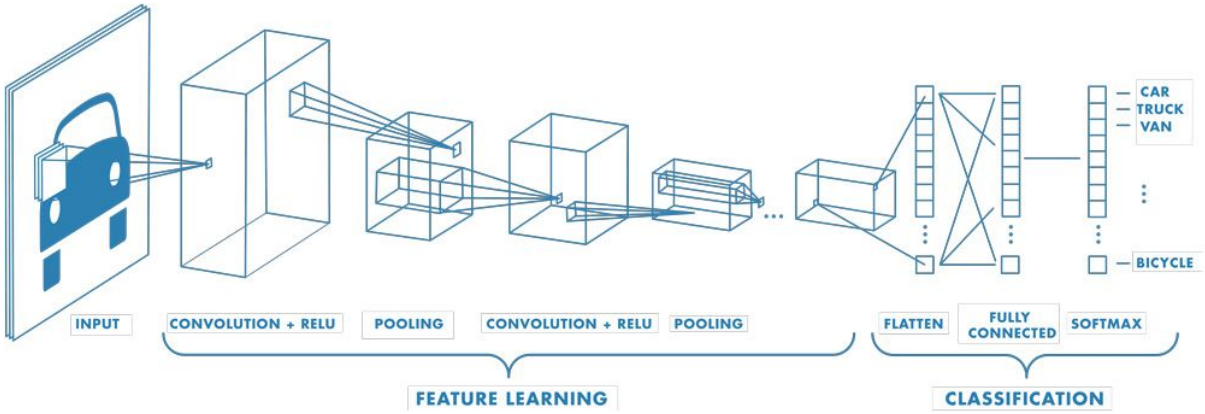


Figure 1.6: A Simple CNN Architecture Example (2)
[17]

2.5 CNNs Different Evaluation Metrics

Evaluation metrics serve as quantitative criteria employed to assess the performance and efficacy of statistical or ML models and provide objective measures to gauge their effectiveness and efficiency. These metrics provide information about model performance and help to compare different models or algorithms.
In classification problems, we use two types of algorithms: class output and probability output. And based on these two, we can divide our evaluation metrics into 2 categories:

1. **Classification Evaluation Metrics:** In the context of a classification task, the primary objective is to predict the target variable, which manifests as discrete values. To assess the performance of such a model, we present some of these metrics below

[18]:

- **Confusion Matrix :** serves as a performance evaluation metric specifically designed for classification problems that involve multiple classes, providing valuable insights into the model's predictive accuracy [14]. It generates a matrix of dimensions N x N, where N represents the total number of classes or categories under consideration for prediction [13]. Figure 1.7 illustrates the full formula :

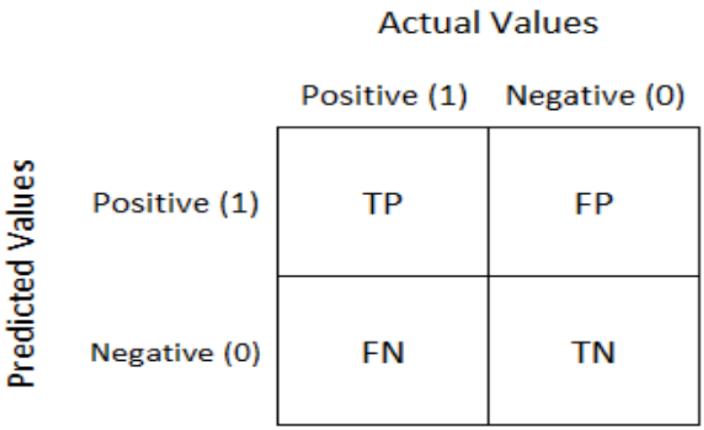


Figure 1.7: Confusion Matrix Formula
[19]

In this context, it is essential to consider the following four terms:
True Positives: instances where the predicted outcome was "Yes" and indeed matched the actual output as "Yes".
True Negatives: instances where the predicted outcome was "No" and indeed matched the actual output as "No".
False Positives: instances where the predicted outcome was "Yes" and indeed matched the actual output as "No".
False Negatives: instances where the predicted outcome was "No" and indeed matched the actual output as "Yes".

- **Classification Accuracy (ACC):** evaluates the frequency at which the classifier accurately predicts the outcomes.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1.1}$$

Where: TP+TN represents the overall count of accurate predictions, while TP+TN+FP+FN indicates the total number of input samples considered for evaluation.
Therefore, accuracy is determined by dividing the number of correct predictions

by the total number of input samples, providing a ratio that represents the accuracy of the model.

- **Logarithmic loss :** Log loss or Cross-Entropy Loss is one of the most common metrics that usually works well with multi-class classification [18]. Right below we illustrate the full formula :

$$logloss = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y_{ij} log(p_{ij}) \tag{1.2}$$

Where: N = the number of rows, M = the number of classes, y = observation's actual value, p = prediction probability.

- **F1 score :** provides a comprehensive assessment by incorporating both precision (accuracy of correctly classified instances) and recall (sensitivity to not miss significant instances). Ranging from 0 to 1, the F1 score reaches its maximum when precision and recall are equal, offering a balanced evaluation [18].

$$F1Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{1.3}$$

Where:

$$Precision = \frac{TP}{TP + FP} \tag{1.4}$$

And:

$$Recall = \frac{TP}{TP + FN} \tag{1.5}$$

- **Area Under Curve (AUC):** curve is a commonly employed metric primarily applicable to binary classification tasks. It quantifies the classifier's capacity to differentiate between classes [18].
- **AUC-ROC :** The ROC curve is a probability curve that depicts the true positive rate (TPR) versus the false positive rate (FPR) at various threshold values [18]. It effectively discriminates between "signal" and "noise" [18], providing a comprehensive evaluation of the classifier's performance.

2. **Regression Evaluation Metrics :** In the context of regression, our objective is to predict the target variable, which manifests as continuous values [18]. To assess the efficacy of such a model, the following evaluation metrics are employed to gauge its performance and effectiveness [18]:

- **Mean Absolute Error(MAE) :** represents the mean discrepancy between the predicted and actual values, providing insights into the accuracy of our predictions in relation to the true output [18]. We illustrate the full formula :

$$MAE = \frac{\sum_{i=1}^{n} |y_i - xi|}{n} \tag{1.6}$$

Where: yi= Prediction, xi = True value,
n = Total number of data points

- **Mean Squared Error(MSE) :** shares similarities with MAE, but it differs in that it computes the squared mean difference between the predicted and actual values [18]. We illustrate the full formula :

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2 \tag{1.7}$$

Where: Yi= Observed values, $\widehat{Y}_i$ = Predicted values,
n = Total number of data points

- **Root Mean Square Error(RMSE) :** is a metric that can be derived by taking the square root of the MSE value [18]. Placing greater emphasis on significant prediction errors due to its higher weightage [18]. We illustrate the full formula :

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}\left\|Y_i - \widehat{Y}_i\right\|^2}{N}} \tag{1.8}$$

2.6 Limitations of CNNs

The main setback in CNNs and any deep learning approach is the term **Explainability** or Explainable-AI. Deep learning is like a black box. No matter how powerful and accurate our models are. We always encounter the usual problem of not having an accurate explanation of why the program makes predictions in a particular state. This lack of explanation has caused companies to think twice about the use of artificial intelligence (AI) in general, particularly in critical areas such as healthcare, finance, and government. In our case, CNNs encounter limitations in comprehending the content within an image. This will take us back to another term which is **Human-in-the-Loop** (HITL). The majority of machine learning models heavily depend on human-prepared data (RSS) [20]. However, the collaboration between humans and machines extends beyond that point. The most robust systems are designed to facilitate ongoing interaction between the two entities, employing a mechanism commonly referred to as "Human in the Loop." [20].
As well as the usual challenges such as classification and detection in complex images, symptoms segmentation, labeled images challenge, and the exploitation of multiple information sources.

2.7 Applications of CNNs

Here are four of the most well-known CNN applications :

- **Healthcare :** CNNs have the capability to examine a multitude of visual reports with the purpose of identifying anomalous conditions in patients [21].
- **Audio Processing :** Keyword detection can be employed in diverse devices equipped with a microphone to identify the occurrence of specific words or phrases when spoken [22].
- **Social media :** social media platforms utilize CNNs to detect and recognize individuals present in a user's profile picture. This application enables users to conveniently tag their friends, enhancing the overall user experience on the platform [21].
- **Object Detection(Automotive) :** CNNs play a vital role in precisely detecting various elements such as traffic signs and objects. This detection capability of CNNs enables automated vehicles to make informed decisions based on the identified objects [22].

3 Genetic Algorithms

As we will deploy the Genetic Algorithm (GA) in our work after recalling some of its principles:

Intuition Behind Genetic Algorithms

The genesis of the Genetic Algorithm (GA) stemmed from intuition and drew inspiration from biological phenomena, where intuition is stating our problem and trying to find an innovative solution for it and biological inspiration is where we started to compose our genetic algorithm.
The impetus for the genetic algorithm derives from Charles Darwin's theory of natural evolution, which embodies the principles of natural selection.

Functional Definition of GA

A genetic algorithm serves as a search heuristic that emulates the mechanism of natural selection, wherein individuals with superior fitness are chosen for reproductive purposes, thus engendering offspring for the subsequent generation. By emulating the principles of natural evolution, this algorithmic approach navigates the solution space, iteratively improving the quality of solutions over successive generations [23].
In essence, it can be perceived as an optimization methodology aimed at identifying the optimal configuration of hyperparameters that yield the utmost precision and accuracy as shown in Figure 1.8:

Figure 1.8: Five Phases of a Genetic Algorithm
[24]

Five Steps Involved in GA

1. **Initialisation:** we establish a population composed of a set of individuals, each representing a potential solution to our problem. These individuals, known as the population, are distinguished by a collection of parameters referred to as genes, encapsulating the key variables for our analysis [24].
   Figure 1.9 represents in the field of biology an analogy that can be drawn with the encoding of genes within a chromosome [23].

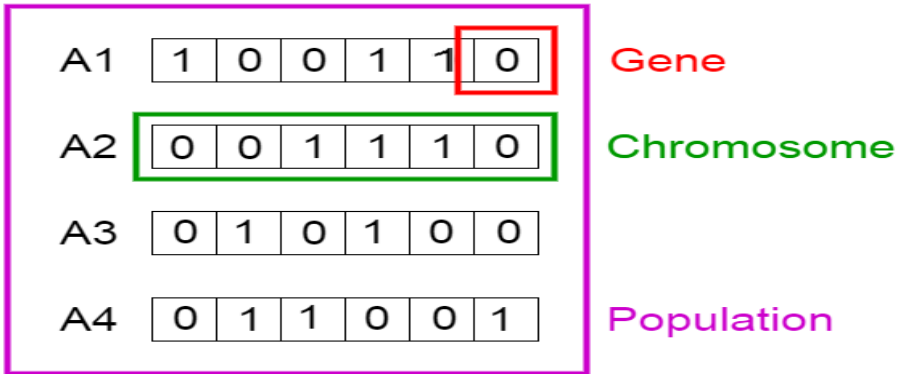


Figure 1.9: Initial Population of a Genetic Algorithm
[24]

.

2. **Fitness Function:** plays a crucial role in evaluating the effectiveness or excellence of an individual solution present within a population. It serves as a guiding metric that aids in the selection and evolution process by assigning a numerical value to signify the fitness or appropriateness of each solution. The probability of an individual being chosen for reproduction in the subsequent generation is determined by their fitness score [23].

3. **Selection** in this phase, we identify and choose the most genetically fit individuals who will pass on their genetic information to the subsequent generation [23]. The selection of individuals is based on their fitness scores where the higher an individual's fitness score the more chances it will be selected for reproduction.

4. **Crossover :** this operation involves the mating of each pair of parents. A crossover point is selected at random within the genetic material (genes) of the parents [23]. Take, for instance, the scenario where the crossover point is identified as 3 as depicted in Figure 1.10, then the genetic material of the parents are interchanged among themselves until the crossover point is attained. This crucial step is commonly referred to as Offspring. Figure 1.11 illustrates the process by applying it to Figure 1.10, and finally, the newly generated offspring are incorporated into the existing population as shown in Figure 1.12. Figure 1.13 illustrates the whole scenario presented above:

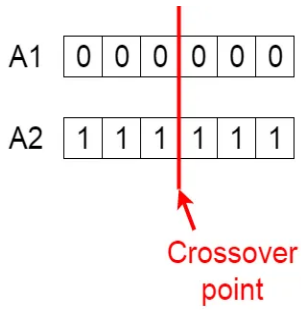


Figure 1.10: Crossover Point
[23]

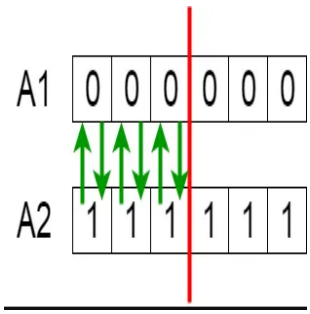


Figure 1.11: Offspring
[23]

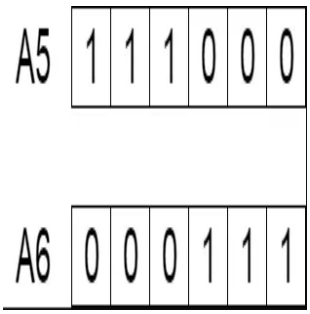


Figure 1.12: Result
[23]

Figure 1.13: Crossover Example
[23]

5. **Mutation :** in order to preserve population diversity and prevent premature convergence, a mechanism is employed where, upon the creation of new offspring, a small random probability is assigned to introduce mutations in certain genes [23]. Figure 1.14 shows an example of mutation:

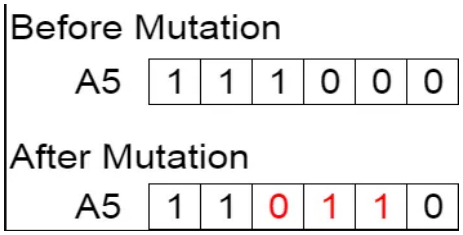


Figure 1.14: Mutation Example [23]

Genetic algorithms (GAs) are search heuristics that can be designed with a termination condition, such as a maximum number of generations, to determine when the algorithm should stop. In our case, the GA terminates when the population converges, meaning that the offspring in a new generation resemble those of the previous generation. This convergence indicates that the GA has derived a set of optimal solutions or hyperparameters for the problem. By utilizing this termination criterion, the GA effectively identifies and converges towards the most suitable solutions, facilitating efficient problem-solving and optimization.

Application of GA

One of the main applications of a genetic algorithm is as an optimization technique in many uses in the commercial world. In addition, it's widely used in Robotics, Engineering Design, Traffic and Shipment Routing (Travelling Salesman Problem), etc.

4 Classification Problems in DL

In the process of designing a CNN, careful consideration must be given to selecting a suitable activation function, how to encode the training data, as well as performance measures tailored to the specific classification or prediction task at hand.
In the realm of ML, classification tasks can generally be classified into three main categories: :

1. **Binary Classification :** When we have only two target classes, we address the object to either the first class or the second for example: Is it Covid-19 in the CXR image or is it a normal case?
2. **Multi-Class Classification :** Within this specific category, we delve into scenarios where classification tasks involve more than two distinct and mutually exclusive targets. In such cases, each input is allocated to a single class, leaving no room for

ambiguity or multiple assignments, for example: Which flower "is" in this picture: dandelion, rose, sunflower, lily, tulip?

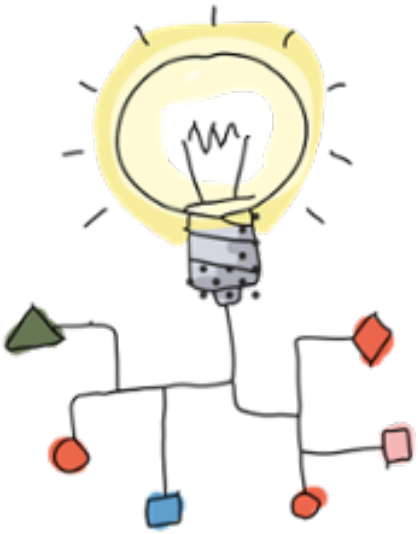
3. **Multi-Label Classification :** Similar to the multi-class classification type, it also deals with scenarios involving more than two classes. However, in the context of multi-label classification, we focus on non-exclusive targets, allowing the possibility of assigning multiple target classes to a single input for example: Which flowers "are" in this picture: dandelion, rose, sunflower, lily, tulip?

Conclusion

In this chapter, we have recalled some preliminaries on DL in order to understand the content of the following chapters. In fact, we introduced A detailed description of some basics of Convolutional Neural Networks (CNNs) and their related metrics also presented, Then we gave an overview of genetic algorithms, and finally presented types of classification problems in machine learning related to our application. While in the next chapter, we will review previous work (Related work) on our topic CXR Analysis Using DL.

Chapter 2

Related Work



In this chapter, we initially identify and list the prominent publicly accessible datasets comprising CXR images that are pertinent to our research, subsequently, we present a comprehensive overview of prevalent approaches employed for preprocessing CXR images, encompassing enhancement techniques and data augmentation strategies. These methods aim to enhance the quality of images and augment the available data, as well as an overview of state-of-the-art deep learning models that leverage CXR images for the detection and diagnosis of various chest diseases. Finally, a comprehensive discussion where we discuss and interpret explainability is provided.
Figure 2.1 illustrates a review of recent advances in deep learning models for chest disease detection using CXR radiography [1]:

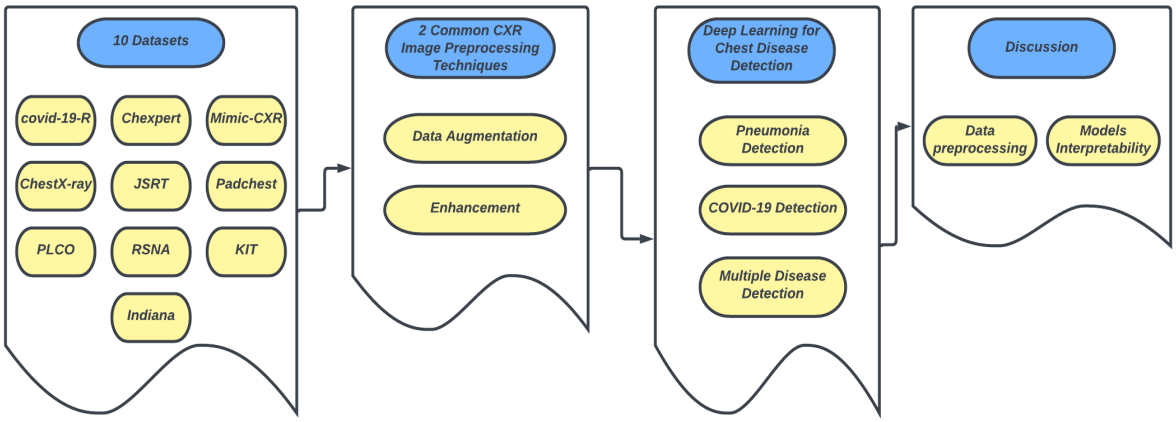


Figure 2.1: Structure of our Related Work
[1]

1 Datasets

In order to choose a dataset to evaluate our DL approach for disease detection we have reviewed the standard datasets built for that purpose.
In the medical field, the domain of our project, we face the problem of diversity and a large number of types of diagnostic methods such as imaging screening, lung biopsy, High-Resolution Computed Tomography (HRCT), and ultrasound imaging. Taking CXR imaging as an example, doctors and specialists use these images to detect diseases and abnormalities, which is not a simple task. This is where using publicly available datasets comes in very handy in deep learning situations.
There are several datasets containing thousands of CXR images. In this section, we focus more on open-access CXR image datasets. Based on our knowledge, we present ten datasets among the most pertinent and widely accessible CXR datasets currently available in the public domain:

1. **COVID-19-Radiography database (Our used dataset)** [1] **:** A collaborative research effort between Qatar University, Doha, Qatar, the University of Dhaka, Bangladesh, and their partners from Pakistan and Malaysia, together with medical professionals, has led to the development of a comprehensive database comprising chest X-ray images. This database focuses on COVID-19-positive cases, as well as normal and viral pneumonia cases, the images encompassed in this dataset were meticulously sourced from various publicly available datasets, online resources, and published literature. The dataset encompasses four distinct labels [2], all images are provided in Portable Network Graphics (PNG) format, with a resolution of 299*299 pixels [9, 10].

2. **CheXpert :** encompasses a substantial collection of 224,316 images obtained from a diverse cohort of 65,240 patients. This dataset was meticulously curated at Stanford Hospital over a span of 15 years, from 2002 to 2017, it encompasses a comprehensive range of 14 common chest abnormalities. To facilitate accurate labeling, each image within the CheXpert dataset has been meticulously annotated for the presence of these abnormalities using an automated rules-based labeler, categorizing them as negative, positive, or uncertain. In order to extract valuable insights from the radiology reports, expert observations have been extracted from the free-text format [1, 25].

3. **MIMIC-CXR :** encompasses a vast assemblage of 377,110 CXR images, meticulously curated to correspond to 227,835 unique patients. Renowned for its expansive scale, it stands as one of the most extensive open-access collections of chest X-rays, complemented by freely accessible radiology reports in a text-based format. This comprehensive dataset encompasses data pertaining to 14 distinct chest abnormalities, making it a valuable resource for research and analysis. The data acquisition process spanned a timeframe between 2011 and 2016, taking place at the esteemed Beth Israel Deaconess Medical Center situated in Boston, MA, and USA [1, 26].

4. **ChestX-ray14 :** represents a compilation of image data meticulously extracted from PACS (Picture Archiving and Communication Systems) databases. This particular dataset serves as an upgraded iteration of the well-known ChestX-ray8 dataset, augmenting it with the inclusion of six additional prevalent chest abnormalities [3], the ChestX-ray14 dataset boasts a substantial collection of 112,120 CXR images, all captured in frontal view. Among these images, 51,708 depict the presence of one or more anomalies, while the remaining 60,412 images exhibit the absence of any of the 14 specified anomalies. The dataset encompasses data derived

[1] Recipient of the esteemed COVID-19 Data Set Award conferred by the Kaggle Community [9, 10]
[2] Including 3616 COVID-19 data images, 10192 normal images, 6012 lung opacity images, and 1345 viral pneumonia images [9, 10].
[3] Namely hernia, fibrosis, pleural thickening, consolidation, emphysema, and edema [1, 27].

from a diverse population of 30,805 unique patients, rendering it a valuable resource for extensive research and analysis [1, 27].

5. **Japanese Society of Radiological Technology (JSRT) :** is a publicly available collection assembled by the JSRT in collaboration with the JRS (Japanese Radiological Society) back in 1998. The dataset was curated by gathering data from 13 medical institutions in Japan and one in the United States. It encompasses a total of 247 posteroanterior CXR images, carefully selected to include diverse cases, among these, 154 images exhibit the presence of nodules, with a breakdown of 100 CXR images containing malignant nodules and 54 CXR images containing benign nodules. Additionally, the dataset includes 93 high-resolution CXR images that serve as a comparison group, devoid of any nodules. This extensive dataset exhibits immense promise for cutting-edge research and meticulous analysis within the domain [1, 28, 29].

6. **Padchest (Pathology Detection in chest radiographs) :** The Padchest dataset stands as a remarkable resource in the field, boasting an expansive collection of meticulously labeled data. It encompasses an impressive assemblage of 168,861 CXR images obtained from a substantial cohort of 67,000 patients who were treated at San Juan Hospital in Spain between 2009 and 2017. The creation of this dataset was a collaborative effort involving the expertise of 18 skilled radiologists, ensuring the accuracy and reliability of the provided annotations [1, 30].

7. **PLCO :** encompasses a substantial collection of 185,241 CXR images, focusing on critical areas [1]. This comprehensive dataset is derived from a cohort of 56,071 male and female patients [1, 31]. The data collection process was conducted within the framework of a comprehensive study examining the influence of cancer screening on mortality rates and secondary outcomes in individuals aged 55-74 years. Notably, the PLCO dataset was meticulously assembled with the generous support and sponsorship of the esteemed National Cancer Institute (NCI) [1, 31].

8. **The RSNA Pneumonia Detection Challenge dataset (RSNA-Pneumonia-CXR) :** curated collaboratively RSNA and the Society of Thoracic Radiology (STR), was specifically compiled and released for an esteemed challenge. This extensive dataset encompasses a collection of 30,000 CXR images, with approximately 15,000 images diagnosed with pneumonia or exhibiting similar pulmonary conditions, such as infiltration and consolidation. Notably, all the images in the RSNA-Pneumonia-CXR dataset have been acquired from the well-known ChestX-ray14 dataset [1, 32].

[1] Prostate, Lungs, Colon, and Ovaries [1, 31].

9. **KIT :** meticulously compiled by the Korea Tuberculosis Association, serves as a significant resource for tuberculosis research [1, 33]. This dataset encompasses a vast collection of 10,848 DICOM images, meticulously collected over a span of several decades from 1962 to 2013 [1] [1, 33].

10. **Indiana :** curated by Demner-Fushman et al., offers a wealth of publicly accessible data for research purposes. This comprehensive dataset comprises a total of 7,470 chest X-ray (CXR) images, encompassing both frontal and lateral views, accompanied by 3,955 associated reports. These data were collected from diverse hospitals and subsequently made available to the esteemed Indiana University School of Medicine. Within this dataset, the CXR images portray a wide spectrum of diseases [2] [1, 34].

Table 2.1 presents a brief description of each dataset:

[1] Within this dataset, there are 7,200 cases classified as normal, representing healthy individuals, while 3,828 cases are classified as tuberculosis cases, portraying instances of the disease [1, 33].

[2] Including pulmonary edema, opacity, cardiac enlargement, and pleural effusion [1, 34]

Table 2.1: Exploring Publicly Accessible CXR Datasets: An Overview [1]

Datasets	Ref	Details	Labels
COVID-19-Radiography database	[9, 10]	21,173 images (299×299 pixels)	Normal, positive COVID-19, opacity, and viral pneumonia
CheXpert	[25]	224,316 images 65,240 patients	14 findings including edema, cardiomegaly, lung opacity, lung lesion, consolidation, pneumonia, atelectasis, pneumothorax, and others
MIMIC-CXR	[26]	473,057 images (2544×3056 pixels) 63,478 patients	14 diseases (227,943 imaging studies)
ChestX-ray14	[27]	112,120 images (1024×1024 pixels) 32,717 patients	14 findings including hernia, consolidation, emphysema edema, pleural thickening, pulmonary fibrosis, and others
JSRT	[28, 29]	247 images (2048×2048 pixels) 247 patients	Nodule and no nodule
Padchest	[30]	160,868 images 67,000 patients	Large number of findings
PLCO	[31]	185,241 images 56,071 patients	Normal and TB
RSNA	[32]	15,000 images	Pneumonia, infiltration, and consolidation
KIT	[33]	10,848 images	Normal and TB
Indiana	[34]	7470 images (512×512 pixels) 3996 patients	Multiple diseases including opacity, cardiomegaly, pleural effusion, and pulmonary edema

2 Common CXR Image Preprocessing Techniques

The initial phase of X-ray image preprocessing encompasses data sampling, formatting, and refinement, with the primary objective of enhancing image quality and ensuring optimal representation.
Along our pre-processing path, we could face many challenges such as compressing data while preserving crucial information within the images, de-identifying patient information for privacy, resizing images while preserving crucial information contained due to the high computing resources required, addressing data imbalances and effectively handling DICOM (Digital Imaging and Communications in Medicine) format, which often encompasses extensive metadata, posing difficulties for non-experts in the field of radiology to interpret and comprehend [1].
Presenting two of the most common pre-processing techniques used on CXR images:

1. **Data augmentation**
   When training a CNN model on an imbalanced dataset, the risk of overfitting arises, resulting in suboptimal outcomes. To address this challenge, augmentation techniques can be employed, offering two primary approaches: position-based augmentation [1] and color-based augmentation [2]. These techniques aim to expand the dataset by introducing subtle modifications to existing images, thereby mitigating the potential for overfitting and enhancing the model's generalization capabilities [1].
   Ait Nasser and Akhloufi [35] incorporated various transformations, including rotation within the range of -15 to 15 degrees, translation by 20% in four directions, shear between 70 and 100, and random flipping, by implementing these techniques, the dataset was augmented, resulting in a total of 84,204 CXR images. This augmentation significantly enhanced the performance of their proposed model [1]. Nayak et al. [36] effectively increased the number of CXR samples by employing diverse techniques such as rotation, scaling, horizontal flipping, and the addition of Gaussian noise with a variance ranging from 0 to 0.25. These augmentation methods yielded a fivefold increase in the number of CXR samples compared to the original training image [1].
   In recent times, there has been a growing interest among researchers in the field of generating synthetic CXR images utilizing Generative Adversarial Networks (GANs) [1]. This approach offers several advantages, including addressing concerns related to patient data privacy and expanding the size of available CXR datasets through the generation of novel artificial images [1]. Venu et al. [37] explored the utilization of GANs for data augmentation on the Pediatric-CXR dataset [38], specifically, they focused on increasing the number of normal CXR images by employing Deep Convolution Adversarial Networks (DC-GAN). Through training the DC-GAN model, they successfully generated high-quality CXR-like images [1]. Chuquicusma et al. [39] leveraged unsupervised DC-GAN to generate realistic images of lung

[1] Such as cropping, rotating, scaling, flipping, padding, etc.
[2] Including adjustments to hue, brightness, contrast, etc [1].

nodules. The authenticity of the generated images was evaluated through Turing tests conducted with two radiologists [1].
Table 2.2 presents a comprehensive examination of various data augmentation methods for CXR images described above:

Table 2.2: A Comprehensive Examination of Various Data Augmentation Methods for CXR Images

Ref.	Datasets	Approach
[35]	Consolidated dataset of 26,316 CXR images from VinDr-CXR and CheXpert datasets	Rotation (-15 to 15 degrees), four directions translation (20%), shear (70 to 100), and a random flip
[36]	703 CXR images from ChestX-ray8 and COVID Chest X-ray and	Rotation, scaling, horizontal flipping, Gaussian noise (variance between 0 and 0.25)
[37]	1341 normal CXR images from Pediatric-CXR	DC-GAN
[39]	ChestX-ray14	Unsupervised DC-GAN

2. **Enhancement**

Typically, image enhancement methods for CXR images are employed to enhance the visibility and clarity of information within the images, catering to both human readers and automated systems. [1].
Aashiq et al. utilized a Gabor filter, which combines Gaussian and sinusoidal elements, to enhance CXR images [1]. In a study conducted by Munadi et al. [40] a combination of adaptive Contrast-Limited Histogram Equalization (CLAHE), high-frequency emphasis filtering, and unsharp masking techniques were employed to enhance tuberculosis (TB) images obtained from the Shenzhen dataset [41] [1], subsequently, the researchers further applied transfer learning using two DCNN models to detect TB using the enhanced images [1]. Tawsifur et al. [10] conducted an investigation on the effects of various enhancement techniques, such as Histogram Equalization (HE), CLAHE, image inversion, and gamma correction, on the performance of DCNN models for COVID-19 detection using CXR images [1]. Nefoussi et al. [42] employed preprocessing techniques like unsharp mask, CLAHE, and HE to enhance CXR pneumonia [1].
Table 2.3 presents a comprehensive examination of various enhancement methods for CXR images described above:

Table 2.3: A Comprehensive Examination of Various Enhancement Methods for CXR Images

Ref.	Datasets	Approach
[1]	COVID-19-Radiography Database	Gabor filtering
[40]	Shenzhen [41]	CLAHE, unsharp masking, and high frequency emphasis filtering
[10]	RSNA-Pneumonia-CXR and BIMCV-COVID19+	HE, CLAHE, image invert, gamma correction, and BCET
[42]	RSNA-Pneumonia-CXR	Unsharp mask, CLAHE, and HE

3 Exploring DL Techniques for Chest Disease Detection Using CXR Images

In this section, we present some of the DL approaches to have a comprehensive examination based on the different types of detection (types of classification) and divided into three categories suiting our study based on the most common diseases chest diseases [1] based on the statistical data provided by the World Health Organization (WHO) [1].

1. **COVID-19 Detection**
   COVID-19 emerged in 2019 in China and was officially declared a global pandemic by the WHO in early 2020 due to its rapid and widespread transmission and severe consequences, in response to this formidable challenge, the utilization of CXR images has been explored for the purpose of detecting COVID-19. The task involved applying DL algorithms for early detection during the initial stages of the pandemic. However, the scarcity of CXR images depicting patients diagnosed with COVID-19 at the onset resulted in the unfortunate loss of numerous lives throughout the duration of the pandemic [1].

   In a recent study conducted by Alqahtani et al. [43] a DL approach was proposed to address the classification task using a dataset consisting of 1504 CXR images. The dataset comprised 504 cases of COVID-19 and 1000 normal cases, and it was collected from the Pediatric-CXR and COVID-19 Chest X-ray datasets. Their method involved the utilization of an Inception-V4 model with transfer learning to accurately detect COVID-19 infections where the performance evaluation of the proposed model yielded an impressive overall accuracy (ACC) of 99.63% [1]. Malathy et al. [44] introduced a DL model named CovMnet, specifically designed to classify CXR images into normal and COVID-19 categories. In the conducted experiments, a comprehensive analysis was carried out involving intricate feature extraction, fine-tuning of CNN hyperparameters, and end-to-end training of four distinct variations of the CovMnet model. Notably, the CovMnet model demonstrated exceptional performance, achieving an impressive ACC rate of 97.40% when applied to CXR images sourced from the Pediatric-CXR dataset [1].In a similar vein, Nguyen et al. [45] trained the DenseNet-121 DCNN model on a substantial collection of 21,165 CXR images obtained from various datasets [2], to enhance the model's performance, they employed HE and a geometric data augmentation technique. This augmentation resulted in the proposed model achieving an impressive ACC rate of 97% [1].
   In Table 2.4 we present the obtained results for different DL studies for COVID-19 detection:

[1] COVID-19, Pneumonia, and multiple diseases
[2] Including COVID-19-Radiography, RICORD, BIMCV-COVID19+, and Pediatric-CXR [1].

Table 2.4: Exploring Various DL pieces of research for COVID-19 Detection: A Comprehensive Overview

Ref.	Datasets	Models	Results
[43]	Custom dataset of 1504 CXR images (504 for COVID-19, and 1000 for normal cases) collected from PediatricCXR and COVID-19 Chest X-ray	Inception-V4 with transfer learning	ACC = 99.63%
[44]	648 CXR images acquired from Pediatric-CXR dataset	Custom DCNN model (CovMnet)	ACC = 97.40%
[45]	COVID-19-Radiography, PediatricCXR, BIMCV-COVID19+, and RICORD	DenseNet-121	ACC = 97%

2. **Pneumonia Detection**
Pneumonia is one of the most common chest diseases and one of the most challenging to detect by radiologists using CXR images as it may be a complication of the flu or any other germs that can cause this condition as well, such as bacteria, viruses, fungi, and others.

Khoiriyah et al. [46] conducted a study where they utilized a dataset consisting of 5856 CXR images from the Pediatric-CXR dataset to perform binary classification of images into normal and pneumonia cases. To enhance the training process, they applied various data augmentation techniques, resulting in an augmented dataset where they trained a personalized CNN model on this augmented data and achieved an impressive ACC of 83.38% in detecting pneumonia [1]. Singh et al. [47] proposed a novel approach using an attention mechanism-based DCNN model for binary classification detecting pneumonia from CXR images, utilizing the Pediatric-CXR dataset. They employed ResNet50 architecture with attention, which yielded remarkable results with an ACC of 95.73% [1]. Darapaneni et al. [48] implemented two distinct DCNN models, namely ResNet-50 and Inception-V4, using transfer learning techniques for binary classification of pneumonia cases. They utilized CXR images from the RSNA-Pneumonia-CXR dataset, where their findings indicated that the Inception-V4 model outperformed ResNet-50, achieving a validation accuracy of 94.00% compared to 90.00% [1].
In Table 2.5 we present the obtained results for different DL studies for Pneumonia detection:

Table 2.5: Exploring Various DL pieces of research for Pneumonia Detection: A Comprehensive Overview

Ref.	Datasets	Models	Results
[46]	Pediatric-CXR	CNN model with and without data augmentation	ACC = 83.38%
[47]	Pediatric-CXR	ResNet50 with attention mechanism	ACC = 95.73%
[48]	RSNA-Pneumonia-CXR	Inception-V4 with transfer learning	ACC = 94%

3. **Multiple Diseases Detection :**
In certain scenarios, patients may encounter the simultaneous occurrence of multiple diseases, posing a significant threat to their well-being. The task of identifying multiple pathologies through CXR images can be challenging for radiologists, primarily due to the resemblances observed among various disease indicators [1].
Wang et al. [27] employed a weakly supervised technique to classify and detect eight lung diseases within the ChestX-ray8 dataset, where their approach yielded promising outcomes, demonstrating an average AUC of 80.30% [1]. Rajpurkar et al. [49] utilized a DenseNet-121 model known as CheXNet, leveraging the ChestX-ray14 dataset for binary classification of the 14 diseases. The CheXNet model achieved notable success, attaining an average AUC of 84.11% [1]. Blais and Akhloufi [50] explored multiple models for detecting lung diseases using the CheXpert dataset. Notably, the Xception DCNN model outperformed other models, delivering an average AUC of 95.87% for six diseases and 94.90% for the 14 diseases encompassed in the dataset [1].
In Table 2.6 we present the obtained results for different DL studies for multiple diseases detection:

Table 2.6: Exploring Various DL pieces of research for Multiple Diseases Detection: A Comprehensive Overview

Ref.	Datasets	Number of Diseases	Models	AUC (Mean)
[27]	ChestX-ray8	8 thoracic diseases	weakly supervised classification method	AUC = 80.30%
[49]	ChestX-ray14	14 thoracic diseases	DenseNet-121 (CheXNet) model	AUC = 84.20%
[50]	CheXpert	14 thoracic diseases	Multiple models (Xception DCNN model used here)	ACC = 94.90%

4 Discussion

In the following section, we direct our attention to the obstacles encountered by the aforementioned studies and deliberate on the significance of model interpretability within the medical domain, an aspect often overlooked by numerous investigations.
Starting with the challenges we might face throughout the project, including data pre-processing, image complexity, symptom segmentation, the challenge of labeled images, leveraging multiple sources of information, etc. For the data preprocessing, we can take the example of collecting well-labeled datasets, which is a process that requires time and high computational resources, moreover, considering that a majority of openly accessible datasets are annotated through automated tag assignment techniques, including keyword-based matching algorithms like those employed in the CheXpert and ChestX-ray14 datasets, or NLP methods such as CheXbert [51] for extracting tags from unstructured radiology reports, it becomes apparent that the outcomes may not meet expectations due to potential errors stemming from factors like inadequate report details [1]. The same logic applied to the other challenges, if we talk about symptom segmentation we'll face a lot of problems because when we focus on DL approaches, we completely forget the main reason we are using this technique and end up ignoring the disease Region Of Interest (ROI) in radiology while applying the segmentation, and if the ROI will be segmented by mistake, that will make our model literally useless.
On the other hand, even when we avoid all the above challenges, the main challenge, and one that is rarely considered by other studies, is the interpretability and explainability of our model. For example, if we take feature extraction as an independent step using different techniques and methods, to provide accurate results in prediction or classification, we will need the intervention of specialists in the field of study of our project to extract relevant features from the input data prior to model training or just diagnose and confirm after our model application [1].

Conclusion

In this chapter, we initially identified and listed the prominent publicly accessible datasets comprising CXR images that are pertinent to our research, subsequently, we presented a comprehensive overview of prevalent approaches employed for preprocessing CXR images, encompassing enhancement techniques and data augmentation strategies. Aiming to enhance the quality of images and augment the available data, as well as an overview of state-of-the-art deep learning models that leverage CXR images for the detection and diagnosis of various chest diseases. Finally, a comprehensive discussion where we discussed and interpreted explainability was provided. While in the next chapter, we will talk in detail about our proposed approach as well as the performance measures that were taken, and the tools and environments used in this work. Finally, we will present the results of the project carried out in our study. Where we will dive into the applied test set, highlight the effects of applying a genetic algorithm to our model, and conclude with a thorough discussion of all of the above.

Chapter 3

Contribution

In this chapter, we provide a comprehensive overview of our proposed methodology, commencing with the meticulous curation of our dataset from the existing pool of publicly accessible resources, the data preparation and pre-processing phase, then diving into our proposed strategy. This involves two steps: the creation of the proposed DCNN architecture as well as the application of a genetic algorithm to refine and tune the hyper-parameters of the targeted problem. The various performance metrics used to evaluate the results are described, as well as the tools and environments used for development. Finally, we present the results of the project carried out in our study. Where we dive into the applied test set, highlight the effects of applying a genetic algorithm to our model, and conclude with a thorough discussion of all of the above.

1 Proposed Approach

In order to deal with disease detection using CXR images, we opt for a CNN-based DL technique and deploy a GA to fine-tune and refine the hyper-parameters of the proposed architecture.

1.1 Dataset Used

In this project, we opted to use a COVID-19-Radiography database [1]. Incorporating a diverse assemblage of CXR images [9, 10]. These images encompass COVID-19-positive cases, along with samples of normal, lung opacity, and viral pneumonia images, the creation of this comprehensive database was a collaborative effort between a consortium of researchers from Qatar University, Doha, Qatar, the University of Dhaka, Bangladesh, and their counterparts from Pakistan and Malaysia, in close collaboration with medical professionals [9, 10].
The compilation of images in this dataset was sourced from a multitude of diverse origins, including various publicly accessible datasets, online repositories, and scholarly publications such as RSNA [32] and Kaggle [52].
Table 3.1 presents a brief description of the dataset:

Table 3.1: Brief Description of COVID-19-Radiography Database

Ref.	Size	Labels	Format	Resolution
[9] [10]	21,165 images 900MB	3616 COVID-19 data images, 10192 Normal images (cases), 6012 Lung opacity images, and 1345 Viral Pneumonia images [9, 10].	PNG	299*299 pixels

[1] Recipient of the esteemed COVID-19 Data Set Award conferred by the Kaggle Community [9, 10]

Data Preparation

In this phase, we need minimal data pre-processing because the data we are using is already formatted, cleaned, and sampled in a way that suited our work.
It may seem like we're avoiding the hard work of pre-processing, but in reality, it will give us unique advantages such as the use of pre-trained models, and save us time to focus more on developing our model and refining the hyperparameters (Finetuning) as we have just done in our approach which will allow us to obtain better results and most importantly organize our workflow. **A few simple changes we made before defining the model:**

- To load all the images, we started with defining the training variable which contains all the images of the training data. We define the shape of the images, resize the images to a resolution of 100*100 pixels, and normalize the images because our CNN relies on gradient calculations;
- The path to each label has been type-pasted (converted) to an array so we can normalize the data (divide it by 255) without having to deal with the Python list limitations;
- Finally, we concatenate all the training data in one training variable using the NumPy library so was can finally divide our data into training and testing.

1.2 The Proposed Synergistic Approach

DCNN models and genetic algorithms are both powerful tools in their own right, but when combined, they have the potential to investigate chest disease detection in a whole new way.
DCNNs excel at pattern recognition and prediction, while genetic algorithms optimize models by refining hyper-parameters and selecting relevant features.
This synergy creates a **feedback loop**, enhancing the accuracy and efficiency of our chest disease detection and potentially saving lives. By leveraging these advanced technologies, we can make significant progress in combating chest diseases.
Figure 3.1 illustrates the concept of our proposed approach:

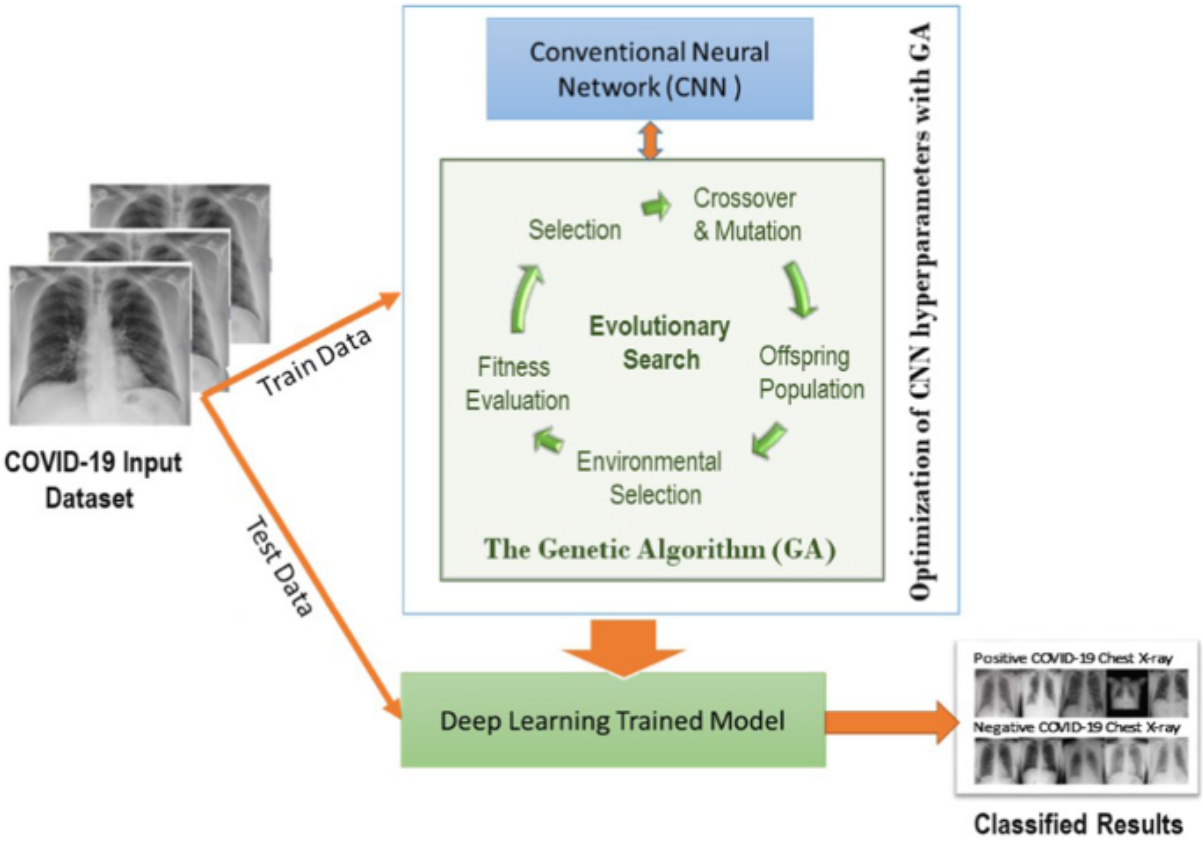


Figure 3.1: The Proposed Synergistic Approach
[53]

1.2.1 Proposed DCNN Architecture

To accomplish the multi-class classification task of CXR images into four distinct categories [1], our research utilizes a specially designed Deep Convolutional Network (DCNN) model. Trained using an extensive dataset consisting of 21,165 CXR images sourced from the COVID-19-Radiography database.
Figure 3.2 below illustrates our proposed DCNN-based architecture:

- The model is built using the Keras [54] DL API: Sequential API [55] which is very simple and easy to use.
- The architectural design of our model incorporates three convolutional layers, with each layer accompanied by a corresponding pooling layer. Additionally, a fully connected layer is employed, followed by three dense layers. The final dense layer is responsible for producing the desired output shape for multi-class classification, utilizing the softmax activation function because of its suitability for multiclass

[1] Covid-19, Pneumonia, Lung Opacity, and Normal Cases

classification.

- The determination of the number of layers within the architecture is **based on empirical analysis and experimentation**. For the other hyperparameters, we will rely on the genetic algorithm.
- We fix the input shape on our input layer and choose the activation function suitable for multi-class classification on the output layer which is softmax in this case with the number of our classes [1].

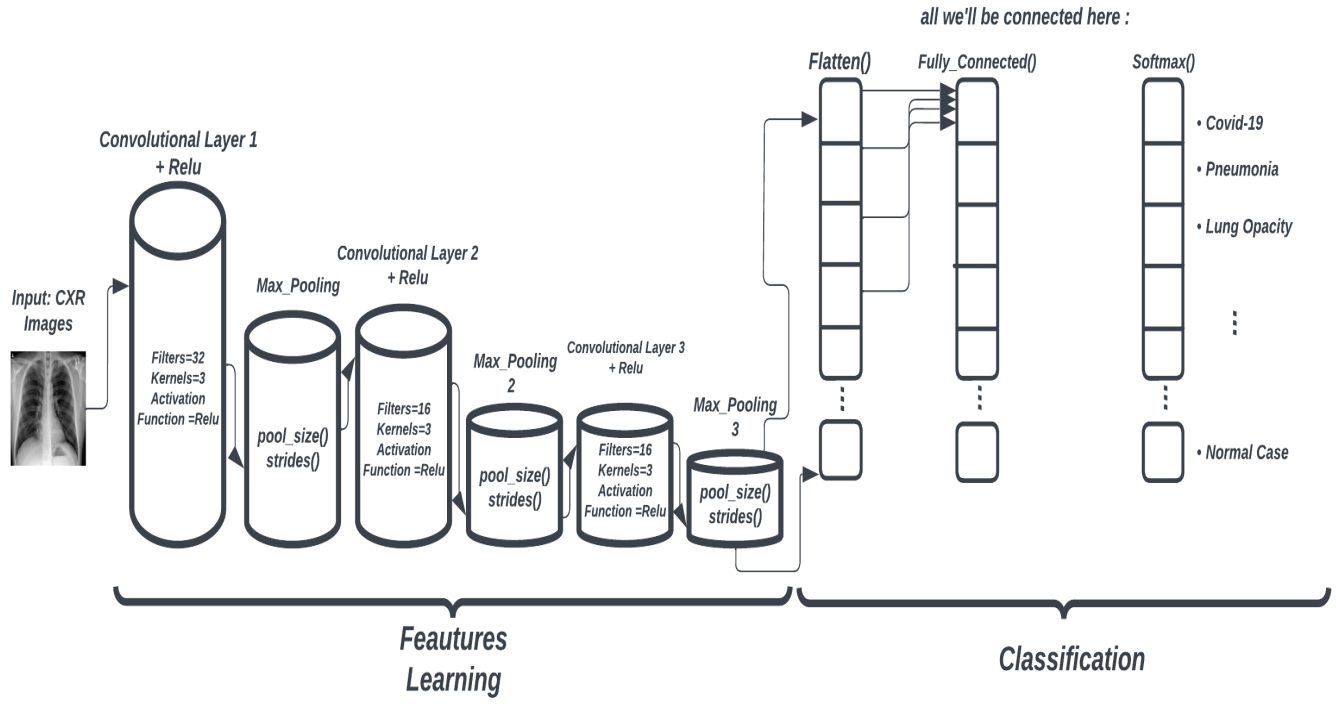


Figure 3.2: Proposed DCNN Architecture Overview

1.2.2 Genetic Algorithm

The PyGAD library, a Python-based open-source tool [11], offers extensive support for various parameters of GAs. These parameters include population size, range and data type of gene values, parent selection techniques, crossover operations, and mutation strategies. Notably, PyGAD serves as a versatile optimization library that grants users the flexibility to customize the fitness function according to their specific needs. The library's utilization entails three primary steps: formulating or defining the fitness function, instantiating the pygad.GA class, and invoking the pygad.GA.run() method. Remarkably, PyGAD extends its support to training DL models generated either within

[1] The 4 labels of our used dataset

the PyGAD framework itself or using popular frameworks like Keras and PyTorch [56].

Within our research, we incorporate a specific module from PyGAD, namely the **pygad.gacnn module**. This module encompasses a class known as pygad.gacnn.GACNN, designed explicitly for training CNNs through the utilization of a GA [57].

An individual within PyGAD-GACNN is encoded using many genes related to the parameters of classical CNN model layers. The targets of our strategy are:

- For the convolutional layer, the targeted parameters or genes in this layer are the number of filters, the size of filters, as well as the number of kernels.
- For the pooling layer, the targeted parameters or genes in this layer are the pool size [1] and the number of strides [2].
- For the dense layer, the targeted parameters or genes in this layer are the number of kernels.

Figure 3.3 illustrates the targeted parameters for the deployment of GA.

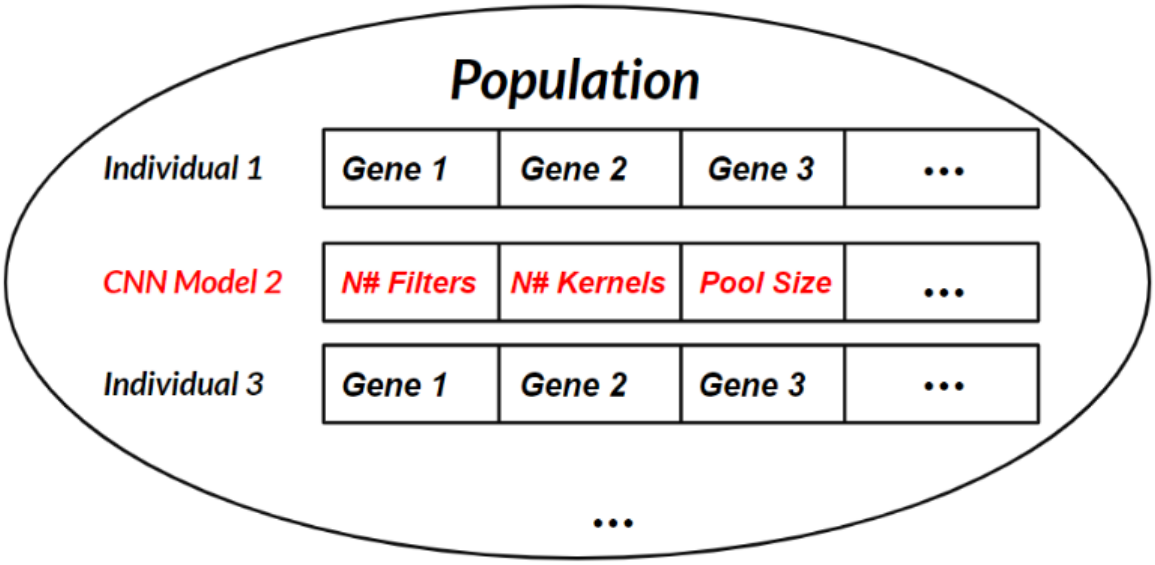


Figure 3.3: Genetic Algorithm: The Targeted Parameters

In the GA framework, the genetic information or parameters of each solution are consolidated into a unified vector representation. Specifically, the weights pertaining

[1] The size of the pooling operation or filter
[2] The number of pixels shifts over the input matrix

to all layers of a CNN are organized into a vector format employing the function "pygad.gacnn.population-as-vectors()" [57]. This vectorized population is subsequently employed as the initial population for further processing [57].
A comprehensive elucidation of the pygad.gacnn.GACNN class, encompassing its constructor, methods, functions, and attributes, can be found within the documentation of the pygad.gacnn Module [56, 57].

2 Deployment of the Genetic Algorithm

In the following section, we provide a detailed account of the procedures involved in utilizing the pygad.gacnn module for constructing and training our CNN employing the GA.
Due to our hardware limitations and even limitations of the computing infrastructure of Google Colab [58] and its sessions that can be maintained for a limited time, we had to apply this genetic algorithm approach to a much simpler proposed DCNN architecture and fewer CXR images than our first DCNN approach to ensure that we end up with results that can be evaluated while ensuring the deployment of a multi-class classification of a balanced set of CXR images of each label from our four labels included in our dataset.

GA-based Construction and Training of CNN using the pygad.gacnn Module: A Step-by-Step Approach

- Similar to building our DCNN, we'll prepare our training data, build our architecture, and create our model.
- Instantiate the pygad.gacnn.GACNN class by providing the necessary arguments which are the model and number of solutions that the genetic algorithm population will have in each generation [57].
- Obtain the population weights in vector form using "the pygad.gacnn.population-as-vectors()" function. This function allows for the consolidation of the weights from each layer of the CNN into a single vector, the resulting population of vectors serves as the initial population for further processing [57].
- Configure the fitness function by preparing the necessary components. Utilize "the pygad.cnn.predict()" function to generate predictions for the class labels using the weights associated with the current solution [57].
- Configure the generation callback function to facilitate the updating of the trained-weights attribute for the layers of the population networks, this function plays a vital role in the training process [57].
- After setting up the GA parameters, you can proceed to instantiate an object of "the pygad.GA class", this class encapsulates the functionalities and operations of the GA [57].

- Execute the instantiated object of "the pygad.GA class" to initiate the execution of the GA, the algorithm will iterate over the specified number of generations as defined by the value provided for the "num-generations" parameter [57].
- Assess the outcomes by generating visualizations of the fitness values and extracting pertinent insights, such as identifying the optimal solution. Additionally, utilize the trained weights to make predictions and compute relevant statistics using conventional metrics [57].

All these steps are explained in full detail in pygad.gacnn Module documentation [57].

During the implementation phase, our study will utilize the GA module to explore and refine the specific hyperparameters of our proposed DCNN model. This process is illustrated in Figure 3.4.

```
sample_shape = data_inputs.shape[1:]
num_classes = 4

data_inputs = data_inputs
data_outputs = data_outputs
```

```
input_layer = pygad.cnn.Input2D(input_shape=sample_shape)
conv_layer1 = pygad.cnn.Conv2D(num_filters=2,
                               kernel_size=3,
                               previous_layer=input_layer,
                               activation_function="relu")
average_pooling_layer = pygad.cnn.AveragePooling2D(pool_size=5,
                                                   previous_layer=conv_layer1,
                                                   stride=3)

flatten_layer = pygad.cnn.Flatten(previous_layer=average_pooling_layer)
dense_layer2 = pygad.cnn.Dense(num_neurons=num_classes,
                               previous_layer=flatten_layer,
                               activation_function="softmax")

model = pygad.cnn.Model(last_layer=dense_layer2,
                        epochs=1,
                        learning_rate=0.01)
```

Figure 3.4: Genetic Algorithm: Implementation

3 Metrics and Used Tools

In order to implement and evaluate our DCNN-based DL approach for disease detection using CXR images, we have deployed some metrics and software.

3.1 Performance Metrics

- **Logarithmic loss :** Log loss (Logistic loss) or Cross-Entropy Loss represents a prominent metric widely utilized for effective evaluation in the context of multi-class classification tasks. We used two types:

  1. **Binary Cross-Entropy**
     Referred to as Binary Log Loss, serves as a prevalent loss function employed within the realm of ML, specifically in scenarios involving binary classification tasks [59]. Its purpose lies in quantifying the disparity between the predicted probability distribution and the actual binary labels associated with a given dataset [59].
     We denote the expressions for binary cross-entropy utilized in binary classification and multi-class classification, respectively:

$$logloss = \frac{1}{n}\sum_{i=1}^{N} -(y_i * log(p_i) + (1 - y_i) * log(1 - p_i)) \tag{3.1}$$

$$logloss = -\frac{1}{N}\sum_{i}^{N}\sum_{j}^{M} y_{ij} log(p_{ij}) \tag{3.2}$$

     Where: N = the number of rows, M = the number of classes, y = observation's actual value, p = prediction probability.

  2. **Categorical Cross Entropy**
     Referred to as Softmax Loss, It encompasses the combination of softmax activation and Cross-Entropy loss, which is commonly employed for multiclass classification tasks [60]. By utilizing this loss function, we can train a CNN to generate a probability distribution across N classes for each input image [60].
     We present the formula for categorical cross-entropy:

$$CCE = -\sum_{i=1}^{N} y_i * log(\widehat{y}_i) \tag{3.3}$$

     Where: N = the number of rows, y = observation's actual value.

- **Classification Accuracy (ACC)**
  The model's performance on the test set is evaluated using the accuracy metric (ACC).
  Where it shows the accuracy of the model or in other words the ratio of total actual predictions to total predictions.

3.2 Tools and Used Environments

In order to implement and evaluate our DCNN-based DL approach for disease detection using CXR images, we deployed a set of open-source tools, languages, and environments. We present them below, justifying their use.

Software Tools

- **Visual Studio Code :** is a freely accessible and robust tool designed for coding purposes, offering a lightweight framework that can be utilized both offline on your personal computer and online via web-based platforms, compatible with various operating systems including Windows, macOS, and Linux [61]. Notably, it features built-in functionality for JavaScript, TypeScript, and Node.js, along with an extensive collection of extensions catering to diverse programming languages supporting multiple runtimes, environments, and cloud services, further expanding its versatility for developers [61].

- **Google Colab :** is a product similar to Jupyter Notebook, offering a convenient environment for Python programmers to write and execute Python code directly within a web browser where developers have the flexibility to experiment with various Python program codes effortlessly [58].

- **Python :** is an interpreted and object-oriented platform with dynamic semantics, boasting high-level built-in data structures, and supports dynamic typing and dynamic binding. These features contribute to its versatility and ease of use in developing software applications [62].

Libraries Used

We have used a lot of libraries, but here are the most commonly used:

- **TensorFlow :** is designed to facilitate numerical computation in a Python environment, enabling efficient ML and streamlined development of neural networks. Helping developers to accelerate their workflow and simplify the implementation of complex algorithms [63].

- **Keras :** developed as a Python-based deep learning API, this platform leverages the power of TensorFlow, a renowned ML framework that facilitates rapid experimentation, providing researchers and developers with a seamless environment to iterate and explore various deep learning models and techniques [54].

- **Scikit-learn :** serves as a valuable asset for practitioners of ML in the Python programming language, offering a wide range of efficient tools, it encompasses various capabilities crucial for ML and statistical modeling tasks enabling users to

effectively analyze and model their data [64].

- **PyGAD:** is designed to facilitate the implementation of GAs and the optimization of ML algorithms. Its functionality encompasses the integration with popular DL frameworks such as Keras and PyTorch [11].

Writing Tools and Environments

- **LaTeX :** stands as a sophisticated typesetting system that encompasses an array of features specifically tailored for the creation of technical and scientific documentation where researchers can ensure the production of high-quality scientific documents that adhere to industry standards and conventions [65].

- **TeXstudio :** serves as an integrated writing environment specifically designed to facilitate the creation of LaTeX documents ensuring a seamless experience for users [66].

- **Overleaf :** is an online platform designed for collaborative writing and publishing, supporting both LaTeX and Rich Text formats. Offering users the possibility to enhance their writing and publishing experience [67].

- **Lucidchart :** is a free-to-sign-up and easy-to-use diagramming web application created by Lucid Software where you can design and even create teams to collaborate on drawing and creating any kind of diagrams, figures, posters, models, and designs, etc. With real-time updates, and allowing you to save your final project in many available formats.

4 Results and Discussion

In this section, we disclose the findings derived from the project undertaken within the scope of our investigation. Where we dive into the applied test set, highlight the effects of applying a genetic algorithm on our model, and end the section with an in-depth discussion of all of the above.

4.1 Applied Test Set

We conducted a series of experiments on the training dataset, training both the proposed technique and the reference model. Subsequently, we evaluated the performance of the models using the validation dataset.

Due to hardware limitations, in addition to our first proposed DCNN architecture, we propose **three DCNN examples with different dataset samples for the deployment of the GA**. All of them have the same architecture and we just modify the number of training data treated in each one by fixing a balanced set of CXR images

of each label from the four labels included in our dataset [1].

1. **The Proposed DCNN sample for the GA (DCNN-1):** 500 CXR images for each label, which will bring our training data to a total of 2000 CXR images.
2. **The Proposed DCNN sample for the GA (DCNN-2):** 600 CXR images for each label, which will bring our training data to a total of 2400 CXR images.
3. **The Proposed DCNN sample for the GA (DCNN-3):** 800 CXR images for each label, which will bring our training data to a total of 3200 CXR images.

Furthermore, we employ the "pygad.gacnn module" to construct and train each of the proposed DCNN architectures, denoted as **GA-DCNN-1, GA-DCNN-2, and GA-DCNN-3** correspondingly.

Training Methodology

The networks in our study were trained using the Adam Optimizer [2]. All models were implemented and trained using the TensorFlow framework. Throughout our experiments, we employed a batch size of 30, enabled verbose mode with a verbosity level of 1, and set the learning rate to 0.1.

Table 3.2 illustrates the evaluation of the different proposed DCNN approaches.

Validation Loss

During the evaluation process, it became evident that our various DCNN approaches exhibited distinct behaviors as they underwent training across multiple epochs. Specifically, the first proposed DCNN architecture demonstrated a notable reduction in validation loss, which decreased from a minimum value of 11.87 to 1.98. While the validation losses of the proposed DCNN samples for the GA (**DCNN-1, DCNN-2, and DCNN-3**) did not improve much and remained constant after reaching a certain epoch.

Validation Accuracy

The ACC parameter describes how efficient and performant our models are. It appears from the evaluations that have been performed that The highest ACC observed among all the sets of approaches utilized is the first proposed DCNN approach, achieving an ACC of 99,53% in 20 epochs. While the most performant proposed DCNN sample for the deployment of the GA among the three proposed is **DCNN-3**, achieving an ACC of 77,17%. After deploying the pygad.gacnn module on each proposed DCNN architecture,

[1] A much simpler proposed DCNN architecture with fewer CXR images than our first DCNN approach each time
[2] a momentum-based gradient descent algorithm incorporating adaptive first-order and second-order moment estimation.

it achieved an ACC of 78,87% with **GA-DCNN-3** in 15 generations.

Table 3.2: Evaluation of Different Proposed DCNN Approaches

Model	*Total Epochs*	Mean Training Time in sec (Epoch)	Classification Accuracy	Test Accuracy	Loss (Cross-Entropy)
Proposed DCNN Trained with the Whole Training Data	5	107,2	95,48%	94,69%	11,87%
	10	109,4	98,51%	95,14%	04,09%
	20	111,15	**99,53%**	95,63%	01,98%
The Proposed DCNN for the GA Deployment (DCNN-1) [1]	5	2,8	**76,80% (-0.51%)**	72%	47,96%
The Proposed DCNN for the GA Deployment (DCNN-2) [2]	5	3,4	**76,67% (-1.56%)**	71,50%	48,30%
The Proposed DCNN for the GA Deployment (DCNN-3) [3]	5	4,4	**77,17% (-1.7%)**	73,75%	44,87%

[1] 500 CXR images for each label, which will bring our training data to a total of 2000 CXR images
[2] 600 CXR images for each label, which will bring our training data to a total of 2400 CXR images
[3] 800 CXR images for each label, which will bring our training data to a total of 3200 CXR images

Table 3.3 illustrates the evaluation results of the GA approach implemented on the proposed DCNN samples, demonstrating its impact on the performance highlighted in blue.

Table 3.3: Evaluation of the GA approach

Model	Number of Solution per Population	Generation	Number of Parents Mating	Mutation Percent of Genes	Classification Accuracy
GA-DCNN-1 [1]	4	10	2	10%	75%
GA-DCNN-2 [2]	4	10	2	10%	74,67%
GA-DCNN-3 [3]	4	10	2	10%	74,41%
GA-DCNN-1 [1]	6	13	3	20%	**77.31% (+0.51)**
GA-DCNN-2 [2]	6	13	3	20%	**78,23% (+1.56%)**
GA-DCNN-3 [3]	6	13	3	20%	**78,87% (+1.7%)**

[1] 500 CXR images for each label, which will bring our training data to a total of 2000 CXR images
[2] 600 CXR images for each label, which will bring our training data to a total of 2400 CXR images
[3] 800 CXR images for each label, which will bring our training data to a total of 3200 CXR images

Figures 3.5, 3.6, 3.7, and 3.8 illustrate the evaluation of our first proposed DCNN along with the three proposed DCNNs with different dataset samples for the deployment of the GA **DCNN-1, DCNN-2, and DCNN-3** respectively.

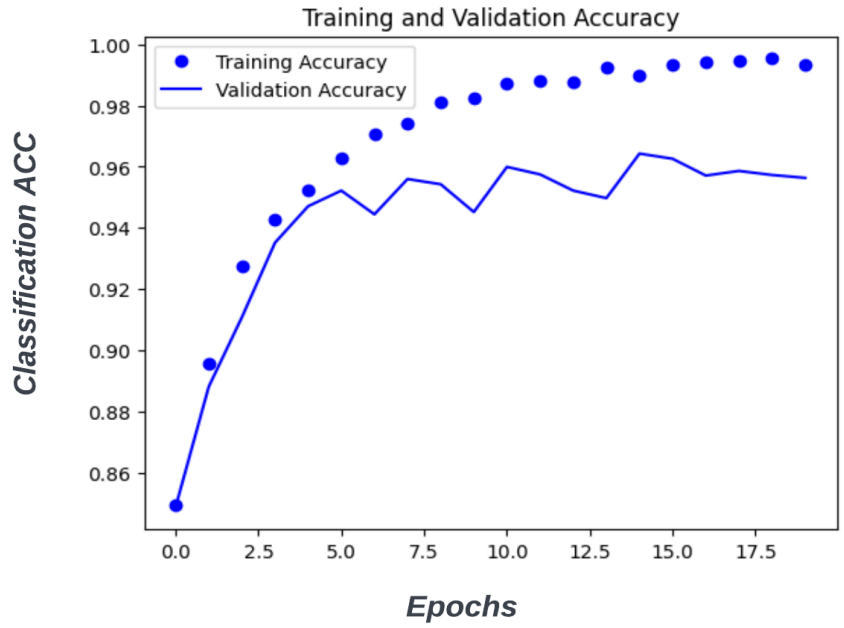


Figure 3.5: Evaluation of the first proposed DCNN

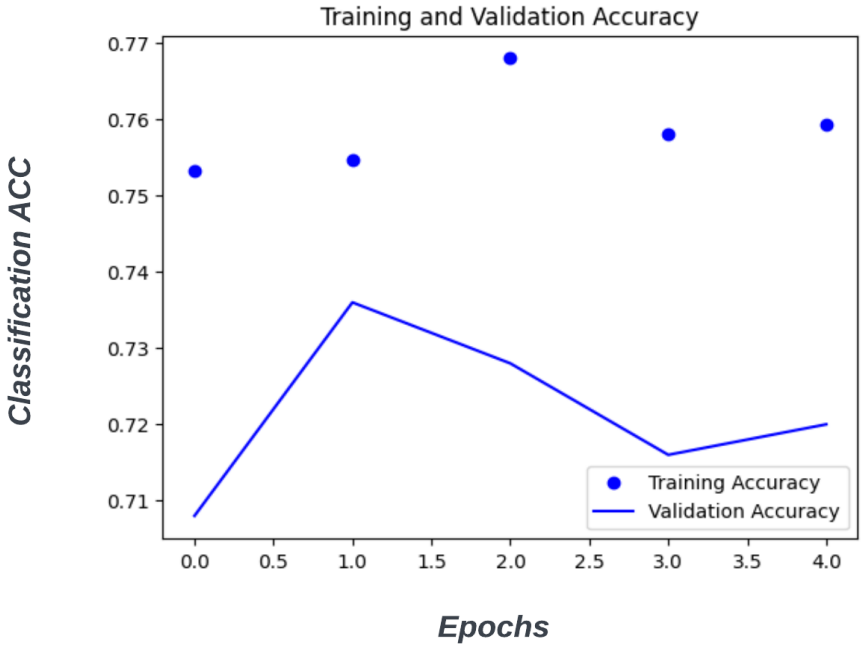


Figure 3.6: Evaluation of the proposed DCNN: **DCNN-1**

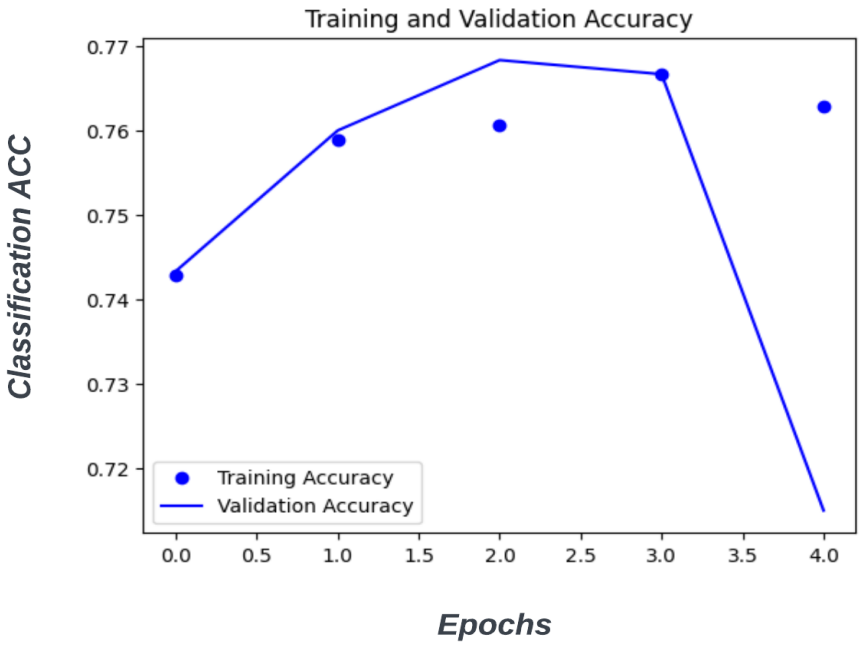


Figure 3.7: Evaluation of the proposed DCNN: **DCNN-2**

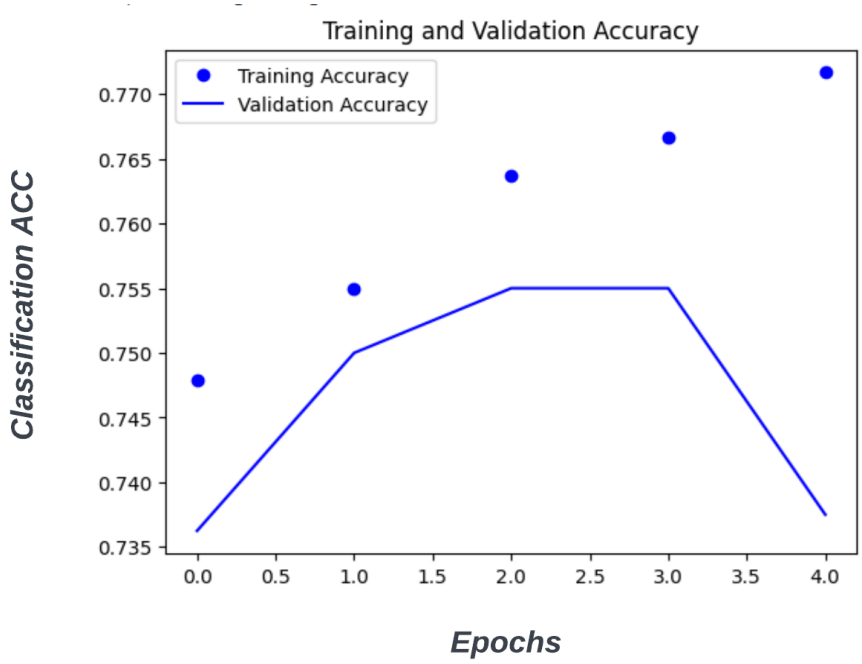


Figure 3.8: Evaluation of the proposed DCNN: **DCNN-3**

To gain further insight into the process, Figure 3.9 showcases an example of the evaluation of the third proposed DCNN sample, referred to as **DCNN-3**, with the deployment of the GA. The figure includes a plot of the fitness score per generation, providing insights into the optimization process. Additionally, it presents details about the best solution, including its index, parameters, fitness value, and the generation at which the best solution was obtained. Furthermore, the figure presents statistics such as the number of correct and wrong classifications, along with the classification accuracy.

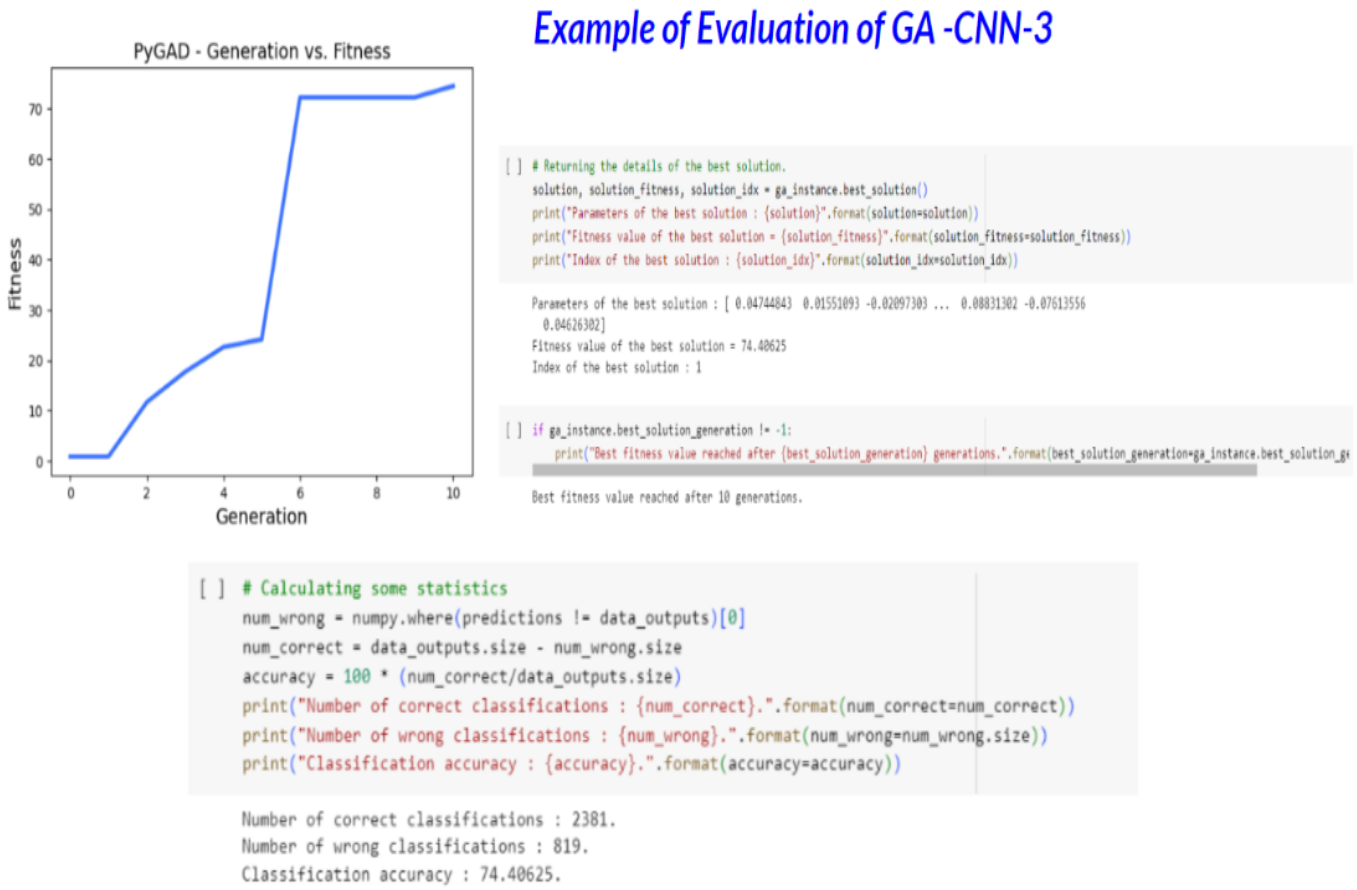


Figure 3.9: Evaluation of **DCNN-3** with the Deployment of the GA: Example

4.2 Effect of Applying a Genetic Algorithm

After the pygad.gacnn module was deployed to train each proposed DCNN samples, we observed an increase of **DCNN-1's** ACC by 0.51%, an increase by 1,56% for **DCNN-2**, and also an increase by 1,7% for **DCNN-3**, compared to the DCNNs without GA deployment. Highlighted in blue in Table 3.2 and 3.3.
Let us recall that due to our hardware limitations, we could no longer increase the parameters of the experiments such as the number of generations, the number of solutions

per population and the number of parental matings, etc., which could have improved the classification accuracy of our **GA-DCNN** models even more.

4.3 Discussion

The results of the study indicate that the initial DCNN model proposed, which underwent training using the complete training dataset (21165 CXR images) performed inherently better and achieved a very high accuracy percentage of 99.53% highlighted in red in Table 3.2. But after applying the GA by deploying the pygad.gacnn module, we observed a slight improvement in the three proposed DCNNs samples. Despite the slight improvement in the performance of the models, by harnessing the capabilities of GAs, we are able to observe the significant impact they have on achieving optimal results. This is achieved through the identification of the most suitable set of hyperparameters and the selection of the fittest individuals from each generation. In fact, with stronger hardware, we are able to maximize the performance of the models and allow the GA to provide the set of solutions with the best hyper-parameters to our problem in order to obtain the most accurate and optimized results.

Conclusion

In this chapter, we provided a comprehensive overview of our proposed methodology, commencing with the meticulous curation of our dataset from the existing pool of publicly accessible resources, the data preparation and pre-processing phase, then dived into our proposed strategy. This involved two steps: the creation of the proposed DCNN architecture as well as the application of a genetic algorithm to refine and tune the hyper-parameters of the targeted problem. The various performance metrics used to evaluate the results were described, as well as the tools and environments used for development. Finally, we presented the results of the project carried out in our study. Where we dived into the applied test set, highlighted the effects of applying a genetic algorithm to our model, and concluded with a thorough discussion of all of the above.

Conclusion and Future Work

Deep Learning has emerged as a highly potent technology for the analysis and processing of CXR images, offering a wide array of methodologies and techniques to aid healthcare professionals in predicting disease risks and facilitating early-stage prevention. This convergence of medicine and computer science has created a promising realm of research, driven by shared interests between the two disciplines. Given the striking similarity in patterns and symptoms among various lung diseases, accurate diagnosis becomes challenging and misinterpretation can have grave consequences. DL models have shown great promise as prediction techniques, achieving precision levels close to human performance. However, the progress of automatic image analysis is hindered by inherent limitations, particularly the lack of explainability or interpretability in DL models.

In this research, a comprehensive examination was carried out to deploy "the pygad.gacnn module" for constructing and training a DCNN using the GA for multi-class classification of CXR images. The focus of our investigation was the COVID-19 radiography database, which comprises a collection of 21,165 chest radiographs. This dataset includes images of COVID-19-positive cases, as well as Normal, lung opacity, and Viral Pneumonia images, making it suitable for our study.

The results demonstrated that the first proposed DCNN model that was trained with the whole training dataset (21165 CXR images) performed inherently better and achieved a very high accuracy percentage. But after applying the GA by deploying "the pygad.gacnn module", we observed the effect of GA by the slight improvement in the three proposed DCNN samples. Hence, it becomes evident that the utilization of the GA allows us to achieve optimal outcomes by identifying the optimal set of hyperparameters and selecting the fittest individuals from each generation. Alternatively, the integration of specialized library modules, similar to the one employed in our investigation, enables

the construction and training of CNN using the GA, facilitating the attainment of accurate results. The evaluation showed that the classification accuracy of our genetic algorithm approach achieved an accuracy (ACC) of 77.31%, 78.23%, and 78.87%. improving it by 0.51%, 1.56%, and 1.7% for the three proposed samples respectively.

In conclusion, in future work, certain perspectives could be integrated to further enrich our scope of work. Such as:

- Add another modality to the input by adding text reports, by applying for example a late fusion strategy such as cross-attention where we will be using the results of our current studies as an extraction of feature representations and add text analysis to process the medical reports associated with them.
- Enlarge the dataset, by extending the processed data with a deeper, more compact dataset like Mimic-CXR;
- Deploy multi-labeling for the classification of multiple disease signs, where a patient can be labeled with more than one disease based on symptoms and according to input CXR images.
- Use high-performance hardware to get more GA efficiency.
- Develop a mobile app and a web app to help people easily enter their CXR images and medical reports and access a specialist who will visually inspect the CXR images for the presence of any disease online.

References

[1] Adnane Ait Nasser and Moulay A Akhloufi. A review of recent advances in deep learning models for chest disease detection using radiography. *Diagnostics*, 13(1):159, 2023. 2, 22, 23, 24, 25, 26, 27, 28, 29, 30, 32, 33, 34

[2] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 2

[3] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 2

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning. *Image Recognition*, 7, 2015. 2

[5] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. arxiv 2016. *arXiv preprint arXiv:1608.06993*, 1608, 2018. 2

[6] Guan-Hua Huang, Qi-Jia Fu, Ming-Zhang Gu, Nan-Han Lu, Kuo-Ying Liu, and Tai-Been Chen. Deep transfer learning for the multilabel classification of chest x-ray images. *Diagnostics*, 12(6):1457, 2022. 2, 3

[7] Xusheng Wang, Cunqi Gong, Mohammad Khishe, Mokhtar Mohammadi, and Tarik A Rashid. Pulmonary diffuse airspace opacities diagnosis from chest x-ray images using deep convolutional neural networks fine-tuned by whale optimizer. *Wireless Personal Communications*, 124(2):1355–1374, 2022. 2

[8] Mohammad Khishe, Fabio Caraffini, and Stefan Kuhn. Evolving deep learning convolutional neural networks for early covid-19 detection in chest x-ray images. *Mathematics*, 9(9):1002, 2021. 2

[9] Muhammad EH Chowdhury, Tawsifur Rahman, Amith Khandakar, Rashid Mazhar, Muhammad Abdul Kadir, Zaid Bin Mahbub, Khandakar Reajul Islam, Muhammad Salman Khan, Atif Iqbal, Nasser Al Emadi, et al. Can ai help in screening viral and covid-19 pneumonia? *Ieee Access*, 8:132665–132676, 2020. 3, 23, 26, 36

[10] Tawsifur Rahman, Amith Khandakar, Yazan Qiblawey, Anas Tahir, Serkan Kiranyaz, Saad Bin Abul Kashem, Mohammad Tariqul Islam, Somaya Al Maadeed, Susu M Zughaier, Muhammad Salman Khan, et al. Exploring the effect of image enhancement techniques on covid-19 detection using chest x-ray images. *Computers in biology and medicine*, 132:104319, 2021. 3, 23, 26, 29, 36

[11] Pygad - python genetic algorithm! URL https://pygad.readthedocs.io/en/latest/index.html. Accessed: 17-06-2023. 3, 39, 45

[12] Shruti M. Differences between ai vs. machine learning vs. deep learning: Simplilearn, Feb 2023. URL https://www.simplilearn.com/tutorials/artificial-intelligence-tutorial/ai-vs-machine-learning-vs-deep-learning. Accessed: 17-06-2023. 6

[13] Manav Mandal. Introduction to convolutional neural networks (cnn). https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/#81b6, Apr 2023. Accessed: 17-06-2023. 7, 8, 10, 12

[14] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015. 8, 9, 12

[15] Phani Ratan. What is the convolutional neural network architecture?, Jan 2021. URL https://www.analyticsvidhya.com/blog/2020/10/what-is-the-convolutional-neural-network-architecture/. Accessed: 17-06-2023. 9

[16] Diego Unzueta. Convolutional layers vs fully connected layers, Mar 2022. URL https://towardsdatascience.com/convolutional-layers-vs-fully-connected-layers-364f05ab460b. Accessed: 17-06-2023. 11

[17] Prabhu. Understanding of convolutional neural network (cnn) - deep learning, Nov 2019. URL https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148. Accessed: 17-06-2023. 11

[18] Evaluation metrics in machine learning, May 2023. URL https://www.geeksforgeeks.org/metrics-for-machine-learning-model/. Accessed: 17-06-2023. 12, 13, 14

[19] Sumeet Kumar Agrawal. Metrics to evaluate your classification model to take the right decisions, May 2023. URL https://www.analyticsvidhya.com/blog/2021/07/metrics-to-evaluate-your-classification-model-to-take-the-right-decisions/. Accessed: 17-06-2023. 12

[20] Arne Wolfewicz. Human-in-the-loop in machine learning: What is it and how does it work?, November 2022. URL https://levity.ai/blog/human-in-the-loop. Accessed: 17-06-2023. 14

[21] Rahul Awati. What are convolutional neural networks?: Definition from techtarget, Apr 2023. URL https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network#:~:text=A%20CNN%20is%20a%20kind,the%20network%20architecture%20of%20choice. Accessed: 17-06-2023. 15

[22] What is a convolutional neural network? 3 things you need to know. URL https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html. Accessed: 17-06-2023. 15

[23] Vijini Mallawaarachchi. Introduction to genetic algorithms - including example code, Mar 2020. URL https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3#:~:text=A%20genetic%20algorithm%20is%20a,offspring%20of%20the%20next%20generation. Accessed: 17-06-2023. 15, 16, 17, 18, 19

[24] Shubham.jain Jain. Introduction to genetic algorithm amp; their application in data science, Jun 2020. URL https://www.analyticsvidhya.com/blog/2017/07/introduction-to-genetic-algorithm/. Accessed: 17-06-2023. 16, 17

[25] Stanford ml group. chexpert a large chest x-ray dataset and competition. URL https://stanfordmlgroup.github.io/competitions/chexpert/. Accessed: 17-06-2023. 23, 26

[26] Alistair EW Johnson, Tom J Pollard, Seth J Berkowitz, Nathaniel R Greenbaum, Matthew P Lungren, Chih-ying Deng, Roger G Mark, and Steven Horng. Mimic-cxr, a de-identified publicly available database of chest radiographs with free-text reports. *Scientific data*, 6(1):317, 2019. 23, 26

[27] Xiaosong Wang, Yifan Peng, Le Lu, Zhiyong Lu, Mohammadhadi Bagheri, and Ronald M Summers. 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*, pages 2097–2106, 2017. 23, 24, 26, 33

[28] Bram Van Ginneken, Mikkel B Stegmann, and Marco Loog. Segmentation of anatomical structures in chest radiographs using supervised methods: a comparative study on a public database. *Medical image analysis*, 10(1):19–40, 2006. 24, 26

[29] Junji Shiraishi, Shigehiko Katsuragawa, Junpei Ikezoe, Tsuneo Matsumoto, Takeshi Kobayashi, Ken-ichi Komatsu, Mitate Matsui, Hiroshi Fujita, Yoshie Kodera, and Kunio Doi. Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules. *American journal of roentgenology*, 174(1):71–74, 2000. 24, 26

[30] Aurelia Bustos, Antonio Pertusa, Jose-Maria Salinas, and Maria de la Iglesia-Vayá. Padchest: A large chest x-ray image dataset with multi-label annotated reports. *Medical image analysis*, 66:101797, 2020. 24, 26

[31] Claire S Zhu, Paul F Pinsky, Barnett S Kramer, Philip C Prorok, Mark P Purdue, Christine D Berg, and John K Gohagan. The prostate, lung, colorectal, and ovarian cancer screening trial and its associated research resource. *Journal of the National Cancer Institute*, 105(22):1684–1693, 2013. 24, 26

[32] The radiological society of north america the society of thoracic radiology. rsna pneumonia detection challenge. URL https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/data. Accessed: 17-06-2023. 24, 26, 36

[33] Sungweon Ryoo and Hee Jin Kim. Activities of the korean institute of tuberculosis. *Osong public health and research perspectives*, 5:S43–S49, 2014. 25, 26

[34] Dina Demner-Fushman, Marc D Kohli, Marc B Rosenman, Sonya E Shooshan, Laritza Rodriguez, Sameer Antani, George R Thoma, and Clement J McDonald. Preparing a collection of radiology examinations for distribution and retrieval. *Journal of the American Medical Informatics Association*, 23(2):304–310, 2016. 25, 26

[35] Adnane Ait Nasser and Moulay A Akhloufi. Chest diseases classification using cxr and deep ensemble learning. In *Proceedings of the 19th International Conference on Content-based Multimedia Indexing*, pages 116–120, 2022. 27, 28

[36] Soumya Ranjan Nayak, Deepak Ranjan Nayak, Utkarsh Sinha, Vaibhav Arora, and Ram Bilas Pachori. Application of deep learning techniques for detection of covid-19 cases using chest x-ray images: A comprehensive study. *Biomedical Signal Processing and Control*, 64:102365, 2021. 27, 28

[37] Sagar Kora Venu and Sridhar Ravula. Evaluation of deep convolutional generative adversarial networks for data augmentation of chest x-ray images. *Future Internet*, 13(1):8, 2020. 27, 28

[38] Daniel S Kermany, Michael Goldbaum, Wenjia Cai, Carolina CS Valentim, Huiying Liang, Sally L Baxter, Alex McKeown, Ge Yang, Xiaokang Wu, Fangbing Yan, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. *cell*, 172(5):1122–1131, 2018. 27

[39] Maria JM Chuquicusma, Sarfaraz Hussein, Jeremy Burt, and Ulas Bagci. How to fool radiologists with generative adversarial networks? a visual turing test for lung cancer diagnosis. In *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*, pages 240–244. IEEE, 2018. 27, 28

[40] Khairul Munadi, Kahlil Muchtar, Novi Maulina, and Biswajeet Pradhan. Image enhancement for tuberculosis detection using deep learning. *IEEE Access*, 8:217897–217907, 2020. 29

[41] Stefan Jaeger, Sema Candemir, Sameer Antani, Yì-Xiáng J Wáng, Pu-Xuan Lu, and George Thoma. Two public chest x-ray datasets for computer-aided screening of pulmonary diseases. *Quantitative imaging in medicine and surgery*, 4(6):475, 2014. 29

[42] Sabrina Nefoussi, Abdenour Amamra, and Idir Amine Amarouche. A comparative study of chest x-ray image enhancement techniques for pneumonia recognition. In *International Conference on Computing Systems and Applications*, pages 276–288. Springer, 2020. 29

[43] Ali Alqahtani, Shumaila Akram, Muhammad Ramzan, Fouzia Nawaz, Hikmat Ullah Khan, Essa Alhashlan, Samar M Alqhtani, Areeba Waris, and Zain Ali. A transfer learning based approach for covid-19 detection using inception-v4 model. *Intelligent Automation & Soft Computing*, 35(2), 2023. 30, 31

[44] Malathy Jawahar, Vinayakumar Ravi, J Prassanna, S Graceline Jasmine, R Manikandan, Rames Sekaran, and Suthendran Kannan. Covmnet–deep learning model for classifying coronavirus (covid-19). *Health and Technology*, 12(5):1009–1024, 2022. 30, 31

[45] Thai Nguyen, Trong-Hop Do, and Quang-Dung Pham. A deep learning based system for covid-19 positive cases detection using chest x-ray images. In *2022 13th International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1082–1087. IEEE, 2022. 30, 31

[46] Septy Aminatul Khoiriyah, Arif Basofi, and Arna Fariza. Convolutional neural network for automatic pneumonia detection in chest radiography. In *2020 International Electronics Symposium (IES)*, pages 476–480. IEEE, 2020. 32

[47] Sukhendra Singh, Sur Singh Rawat, Manoj Gupta, BK Tripathi, Faisal Alanzi, Arnab Majumdar, Pattaraporn Khuwuthyakorn, and Orawit Thinnukool. Deep attention network for pneumonia detection using chest x-ray images. 2023. 32

[48] Narayana Darapaneni, Ashish Ranjan, Dany Bright, Devendra Trivedi, Ketul Kumar, Vivek Kumar, and Anwesh Reddy Paduri. Pneumonia detection in chest x-rays using neural networks. *arXiv preprint arXiv:2204.03618*, 2022. 32

[49] Pranav Rajpurkar, Jeremy Irvin, Kaylie Zhu, Brandon Yang, Hershel Mehta, Tony Duan, Daisy Ding, Aarti Bagul, Curtis Langlotz, Katie Shpanskaya, et al. Chexnet: Radiologist-level pneumonia detection on chest x-rays with deep learning. *arXiv preprint arXiv:1711.05225*, 2017. 33

[50] Marc-André Blais and Moulay A Akhloufi. Deep learning and binary relevance classification of multiple diseases using chest x-ray images. In *2021 43rd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, pages 2794–2797. IEEE, 2021. 33

[51] Akshay Smit, Saahil Jain, Pranav Rajpurkar, Anuj Pareek, Andrew Y Ng, and Matthew P Lungren. Chexbert: combining automatic labelers and expert annotations for accurate radiology report labeling using bert. *arXiv preprint arXiv:2004.09167*, 2020. 34

[52] Paul Mooney. Chest x-ray images (pneumonia), Mar 2018. URL https://www.kaggle.com/paultimothymooney/chest-xray-pneumonia. Accessed: 17-06-2023. 36

[53] Md Rafiul Hassan, Walaa N Ismail, Ahmad Chowdhury, Sharara Hossain, Shamsul Huda, and Mohammad Mehedi Hassan. A framework of genetic algorithm-based cnn on multi-access edge computing for automated detection of covid-19. *The Journal of Supercomputing*, 78(7):10250–10274, 2022. 38

[54] Keras Team. Keras documentation: About keras. URL https://keras.io/about/. Accessed: 17-06-2023. 38, 44

[55] Soumyaranjan Swain. Understanding sequential vs functional api in keras, Aug 2021. URL https://www.analyticsvidhya.com/blog/2021/07/understanding-sequential-vs-functional-api-in-keras/. Accessed: 17-06-2023. 38

[56] Ahmed Fawzy Gad. Pygad: An intuitive genetic algorithm python library. *arXiv preprint arXiv:2106.06158*, 2021. 40, 41

[57] pygad.gacnn module. URL https://pygad.readthedocs.io/en/latest/gacnn.html. Accessed: 17-06-2023. 40, 41, 42

[58] Tamal Das. Google colab: Everything you need to know, Aug 2022. URL https://geekflare.com/google-colab/. Accessed: 17-06-2023. 41, 44

[59] Shipra Saxena. Binary cross entropy/log loss for binary classification, May 2023. URL https://www.analyticsvidhya.com/blog/2021/03/binary-cross-entropy-log-loss-for-binary-classification/#h-what-is-binary-cross-entropy-or-logs-loss. Accessed: 17-06-2023. 43

[60] Shah Deval. Cross entropy loss: Intro, applications, code, Jan 2023. URL https://www.v7labs.com/blog/cross-entropy-loss-guide#:~:text=Categorical%20Cross%20Entropy%20is%20also,N%20classes%20for%20each%20image. Accessed: 17-06-2023. 43

[61] Martin Heller. What is visual studio code? microsoft's extensible code editor, Jul 2022. URL https://www.infoworld.com/article/3666488/what-is-visual-studio-code-microsofts-extensible-code-editor.html. Accessed: 17-06-2023. 44

[62] What is python? executive summary. URL https://www.python.org/doc/essays/blurb/. Accessed: 17-06-2023. 44

[63] Serdar Yegulalp. What is tensorflow? the machine learning library explained, Jun 2022. URL https://www.infoworld.com/article/3278008/what-is-tensorflow-the-machine-learning-library-explained.html. Accessed: 17-06-2023. 44

[64] Kunal Jain. Scikit-learn(sklearn) in python - the most important machine learning tool i learned last year!, Jun 2020. URL https://www.analyticsvidhya.com/blog/2015/01/scikit-learn-python-machine-learning-tool/. Accessed: 17-06-2023. 45

[65] Latex – a document preparation system. URL https://www.latex-project.org/#:~:text=LaTeX%20is%20a%20high%2Dquality,is%20available%20as%20free%20software. Accessed: 17-06-2023. 45

[66] Welcome to texstudio. URL https://www.texstudio.org/. Accessed: 17-06-2023. 45

[67] Overleaf: Collaborative writing and publishing, Dec 2022. URL https://www.digital-science.com/product/overleaf/. Accessed: 17-06-2023. 45