Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 9, No.2, September : 2024

CLASSIFICATION AND DETECTION OF COVID-19 USING GENERATIVE ADVERSARIAL NETWORK

Rasamalla Vishwasree, PG Scholar, Department of Electronics And Communication Engineering, University College of Engineering Jagtial Jawaharlal Nehru Technological University Hyderabad.
vishwasareerasamalla076@gmail.com
B. Prabhakar, Professor and Head, Department of Electronics And Communication Engineering, University College of Engineering Jagtial Jawaharlal Nehru Technological University Hyderabad,

Abstract
Covid-19 is accountable for five out of the thirty leading causes of mortality worldwide, presenting a significant peril to individuals' well-being and existence. Timely and precise diagnosis is crucial for speeding up recovery and improving long-term chances of survival. Deep learning algorithms has substantial potential for the automated, expeditious, and precise identification of pulmonary diseases in medical images. Convolutional neural networks possess demonstrated potential in the field of disease diagnosis. However, the efficacy of these supervised models greatly the outcome relies on the existence of significant quantity of annotated data, which is costly and challenging to gather, especially for a recently identified illness. This study presents a comprehensive unsupervised framework that efficiently categorizes Covid-19 illness by leveraging chest CT and X-ray data. Our method offers Lung-GANs, which are advanced generative adversarial networks capable of producing significant Utilizing unannotated data to depict images of lung disease. We utilize the characteristics of the lung build a SVM model and a Stacking Classifier model. Our research has shown that the proposed methodology outperforms the existing leading unsupervised models in accurately classifying lung illnesses. The Yolov5 and Yolov8 models are employed for the identification of COVID-19. The model demonstrated exceptional performance across all six comprehensive measures, publicly accessible lung disease datasets that were analyzed in this study. Hence, the suggested approach will enhance the categorization and identification of COVID-19 conditions by decreasing the time required for diagnosis and augmenting overall effectiveness.

Keywords: Covid 19 , Deep Learning , Testing .UML Diagrams , Test cases , Class Diagrams

Introduction
Pulmonary illnesses are among the primary factors contributing to death and impairment on a global scale. The most common respiratory diseases are COVID-19, tuberculosis (TB), and pneumonia. Prompt identification is necessary to expedite recuperation and improve the likelihood of survival. In the past, lung disease was typically diagnosed using chest radiography and computed tomography scans. Chest X-rays are more cost-effective, less labor-intensive, more widely accessible, and have a shorter duration compared to CT scans. Even experienced radiologists struggle to differentiate between identical tumors and detect tiny nodules. In addition, the process of manually screening for pulmonary illnesses is characterized by being time-consuming, requiring a significant amount of effort, and being susceptible to differences in interpretation between different observers as well as within the same observer.
Moreover, a scarcity of radiologists coupled with a rise in pulmonary infections could potentially lead to a postponement in the identification and management of the condition. Developing a highly efficient A computerized system designed to assist in the diagnosis of lung diseases during the screening process is of utmost importance. This system should aim to decrease diagnosis times and improve diagnostic convenience. This is especially important considering the ongoing COVID-19 pandemic crisis. This motivated us to develop a system that could assist in the identification of lung disorders by Empowering skilled Radiologists should alleviate to manage their tasks and assist novice doctors

in delivering prompt and precise evaluations.
Deep convolutional GANs, a more recent iteration of GANs, have exhibited promising outcomes in tasks related to image synthesis, showcasing the ability of GANs to effectively learn representations of images and capture intricate data distributions. In this study, we provide a sophisticated unsupervised approach to acquire meaningful representations of lung illness images. Our method relies solely on unlabeled data and employs a generative adversarial network model.

Software Implementation
The Python software ecosystem comprises a variety of tools and methodologies that are specifically designed to streamline the process of developing and deploying Python applications in a highly efficient manner. Essentially, it comprises of the Python interpreter (such as C Python), integrated development environments (such as PyCharm or VSCode), and package management tools (such as pip). Virtual environments, established using technologies such as venv or virtualenv, guarantee segregated and uniform project configurations. The environment incorporates specialized There are various libraries and frameworks specifically created for distinct areas of expertise. Two popular frameworks for Two popular frameworks for web development are Django and Flask. NumPy and Pandas are crucial libraries for data science. And for machine learning, there are TensorFlow and PyTorch. In addition, version control technologies such as Git and deployment platforms like AWS or Docker improve collaboration and simplify the deployment process. This ecosystem offers a well-organized and adaptable framework for creating resilient Python applications.

Deep learning
Deep learning is a specialized branch of machine learning that focuses on neural networks with three or more layers. These neural networks attempt to imitate the functions of the human brain, but they do not reach the same level of performance. They have the capacity to gather information from extensive databases. While a neural network with a single layer can offer approximate predictions, the addition of additional hidden layers can improve optimization and enhance accuracy.

Python language
Python is an interpreted, object-oriented, and high-level programming language. The system possesses dynamic semantics. The language's sophisticated existing data structures, together with its ability to automatically deduce variable types and form associations between variables and objects at runtime, make it extremely attractive for Rapid Application Development. Moreover, it is very suitable for utilization as a scripting or adhesive language to establish connections between pre-existing components. The Python syntax is clear and easily comprehensible, placing emphasis on legibility and eventually reducing costs associated with software upkeep. Python supports the utilization of modules and packages, which facilitates the structuring of programs into smaller, reusable elements.

System Testing
System testing, also known as system-level tests or system-integration testing, is the evaluation conducted by a quality assurance team to examine the interaction between different components of an application inside the overall, unified system or program. System testing is the procedure of confirming that a software fulfills its intended design by executing its features. Blackbox testing is a method that involves a comprehensive evaluation of an application's functionality without any knowledge of its internal structure or code. System testing guarantees the delivery of the intended outcome for all types of user input throughout the whole program.

Phases of system testing
This video tutorial offers a comprehensive showcase of the systematic strategies and procedures

required to successfully overcome this specific level of the test. System testing is a thorough assessment of all the elements of a program to verify their proper operation as a unified entity. System testing is often performed by a Quality Assurance team following the functional or user-story testing of individual modules and the integration testing of each component.. After the completion of system testing and the attainment of the desired outcomes, a software build is subjected to acceptance testing as a conclusive assessment prior to its deployment in production. Users utilize the application during the production process. A team of application developers meticulously records every instance of problems and sets the permissible categories and quantity of difficulties.

Software Testing Strategies

The optimal technique is the most successful strategy for enhancing testing efficiency in software engineering. A software testing plan delineates the precise steps, schedule, and techniques necessary to guarantee the creation of a superior end product. Usually, the main objective is accomplished by utilizing various combinations of software testing methodologies:

Static Testing

The initial testing approach utilized is static testing, which entails assessing the evolving product without executing it. Desk-checking is an essential process used to find bugs or errors inside the code. This analysis is of this type. Conducting code review during the pre-deployment phase is crucial to identify and resolve any issues that may result from coding errors and vulnerabilities in the program's architecture.

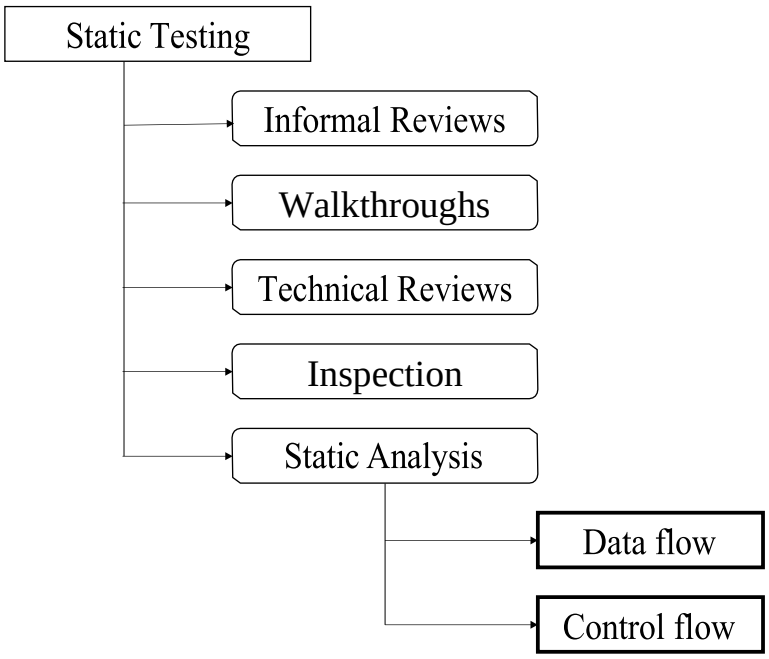


Figure 1: static testing

Structural Testing

Assessing software without executing it is not a practical option. Structural testing, also known as white-box testing, is essential for detecting and rectifying faults and errors that occur during the pre-production stage of software development. During this phase, regression testing is performed to carry out unit testing in alignment with the program's architecture. Usually, an automated technique functions within the test automation framework to accelerate the development process at this stage. Developers and QA engineers have extensive access to the software's architecture and data flows, allowing them to detect any changes in the system's behavior by comparing test results with previous versions.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 9, No.2, September : 2024

Types of structural Testing

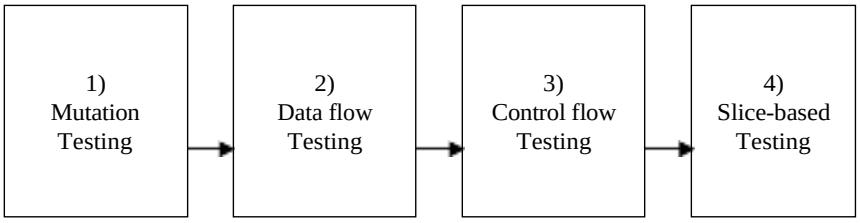


Figure 2: structural testing

Behavioral Testing

The ultimate round of testing mostly centers on assessing the software's responses to diverse stimuli

Black Box Testing

.The focus is on the actions themselves, rather than the underlying systems that control these responses. Psychometric assessment also referred to as black-box testing, is the systematic evaluation of a product from the user's standpoint through the execution of several tests, predominantly employing manual techniques. QA engineers possess specialized knowledge about a company or the intended features of a software, referred to as The Blackbox, which enables them to conduct usability tests and identify and fix defects in a manner similar to ordinary users of the product. Behavioral testing can utilize automation, specifically for regression tests, in order to mitigate the potential for human error when performing repetitive tasks. For instance, it may be important to complete The website requires 100 registration forms to evaluate the effectiveness of the product in handling this task.. Therefore, it is recommended to automate this test.

Figure 3: behavioral testing

Table 1: Test cases

S.NO	INPUT	If available	If not available
1	User signup	User get registered into the application	There is no process
2	User sign in	User get log in into the application	There is no process
3	Enter input for prediction	Prediction resultsdisplayed	There is no process

Proposed Method

This project presents a sophisticated unsupervised framework for categorizing lung illnesses based on chest CT and X-ray images. Our approach presents Lung-GANs, which are multi-layer generative adversarial networks that acquire understandable Generating visual depictions of lung illness images exclusively from unannotated input. The lung characteristics obtained from the model are used to Perform training on a Support Vector Machine (SVM) with Stacking Classifier. We perform comprehensive assessments and experiments on the suggested approach utilizing six extensive publically accessible datasets consisting of chest X-ray and CT pictures. We assess the efficacy of Lung-GANs in classifying lung disorders using visual analysis of the acquired features.

CNN

A Convolutional Neural Network is a specialized type of neural network designed primarily for tasks

involving picture categorization and recognition. Convolutional neural networks are widely employed in various fields, including scene labeling, object detection, and face recognition. The CNN algorithms employ image analysis and categorization techniques To categorize an input image into different classes unique categories, such as dog, cat, lion, tiger, and others. A computer processes an image by interpreting it as a grid of pixels and relies on the image's resolution. The image resolution can be calculated by multiplying the height (h), width (w), and dimension (d). An RGB image is denoted as a 6 * 6 * 3 matrix, denoting its size of 6 rows, 6 columns, and 3 color channels. Conversely, a grayscale image is denoted by a matrix with dimensions 4 * 4 * 1, signifying that it possesses 4 rows, 4 columns, and just 1 color channel. In the CNN architecture, every input image undergoes a series of Convolutional layers, pooling procedures, and fully linked layers. These layers are improved by applying filters, commonly referred to as kernels. Next, we will utilize the SoftMax approach to categorize an item based on probabilistic values of 0 and 1.

Algorithms

1. GAN Model - Generative adversarial networks are a cutting-edge Progress in the domain of machine learning. GANs are a category of generative models that possess the capability to produce novel data instances that closely resemble the data used for training. Generative Adversarial Networks can produce images that closely resemble photographs of human faces, even if these faces are not representative of any real humans. The Linear SVC algorithm is a technique that seeks to find a hyperplane that maximizes the distance between detected samples.
2. The Random Forest algorithm, created by Leo Bierman and Adele Cutler, is a widely used and well-regarded machine learning technique. It aggregates the outcomes of numerous decision trees to obtain a singular outcome. The tool's remarkable ease of use and flexibility have greatly contributed to its widespread appeal. It efficiently addresses both classification and regression difficulties.
3. A Voting Classifier is a machine learning model that use an ensemble of many models to generate predictions. The function chooses the output that has the greatest likelihood among the given classes.
   4.YoloV5 employs a unique method called "dynamic anchor boxes" to create the anchor boxes. The approach involves using a clustering algorithm to classify the ground truth bounding boxes into clusters, and then using the centroids of these groups as the anchor boxes.
4. YoloV8-YOLOv8 is the latest version of the YOLO algorithm. The newer versions of the product outperform previous versions by integrating improvements like as spatial attention, feature fusion, and context aggregation modules.

System architecture

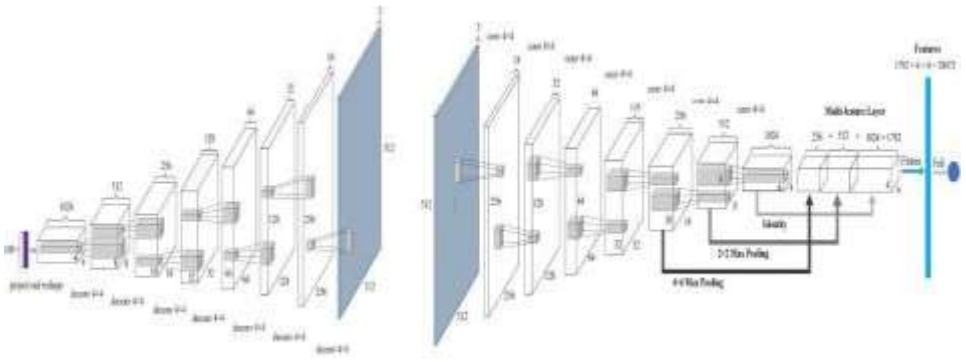


Figure 4: system architecture

Data Flow Diagram

1.Data transfer A diagram is alternatively referred to as a bubble chart. The graphical formalism is a method that employs visual representations to illustrate a system, encompassing the input data, the diverse processing processes applied to this data, and the resultant output data generated by the system.

2.The data flow diagram is an essential modeling tool. It is employed to replicate the system components. The system consists of four essential components: the system process, the data used by the process, an external entity that interacts with the system, and the information flows inside the system.

3.Data flow refers to the movement of data from one location to another. The diagram illustrates the flow of data within a system and the subsequent modifications it undergoes through a sequence of transformations. The figure depicts the process of transmitting information and the requisite alterations that take place from the initial input to the final output.

4. A Data Flow Diagram is commonly known as a bubble chart. A Data Flow Diagram is a visual depiction that demonstrates the movement of data within a system. It serves as a representation of a system at any degree of abstraction. Data Flow Diagrams can be classified into many levels according to their escalating intricacy and amount of data.

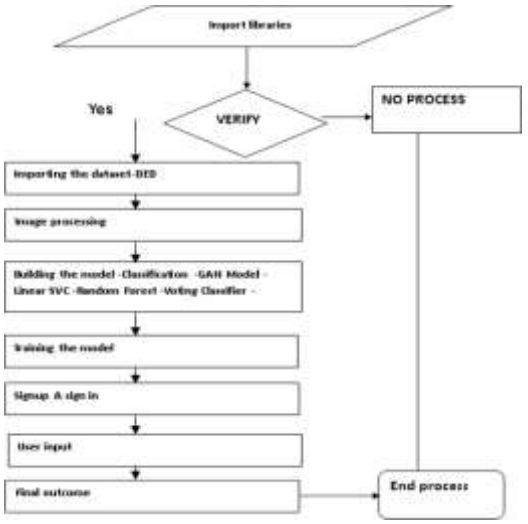


Figure 5: dataflow diagram of classification detection of covid-19

UML Diagrams

UML is an acronym for Unified Modeling Language. UML is a widely respected and commonly used modeling language in the domain of object-oriented software engineering. The Object Management Group is responsible for supervising and establishing the standard. The objective is to establish UML as a globally acknowledged language for constructing models of object-oriented computer software. The UML framework consists of two main components: a Meta-model, which establishes the essential concepts and interactions inside UML, and a notation, which visually expresses these concepts and relationships. In the future, UML will be able to integrate several approaches or strategies. The Unified Modeling Language is a standardized language used for the purpose of planning, visualizing, building, and documenting software system components. Moreover, it is employed in the realm of business modeling and other non-software systems. The UML is a robust framework that utilizes well-established technical ideas to precisely represent intricate and inclusive systems. The Unified Modeling Language is important for the creation of object-oriented software and the efficient administration of the complete software development process. The UML predominantly utilizes visual symbols to depict the design of software projects.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 9, No.2, September : 2024

USE CASE DIAGRAM

A UML use case diagram is a behavioral diagram that is created by analyzing use scenarios. The main objective of the system is to efficiently showcase its operation, which includes presenting individuals and their various goals. The essay delineates the diverse circumstances in which the usage occurs and scrutinizes any possible connections that may exist between these scenarios. A use case diagram is commonly employed to depict the distinct system functionalities executed by each user involved. The actions performed by the actors within the system can be depicted.

Figure 6: use case diagram

CLASS DIAGRAM

The class diagram complements the information shown in the use case diagram to establish a full blueprint of the system's design. The class diagram organizes the players identified in the use case diagram into a set of interconnected classes. The relationship between the classes can be classified as either a "is-a" or "has-a" relationship. Every class shown in the class diagram has the ability to provide distinct and precise functionality. The class offers a diverse range of functions. The class's "methods" are frequently referred to as "methods". Moreover, each class possesses the capacity to have unique "attributes" that serve as distinctive traits for the class.

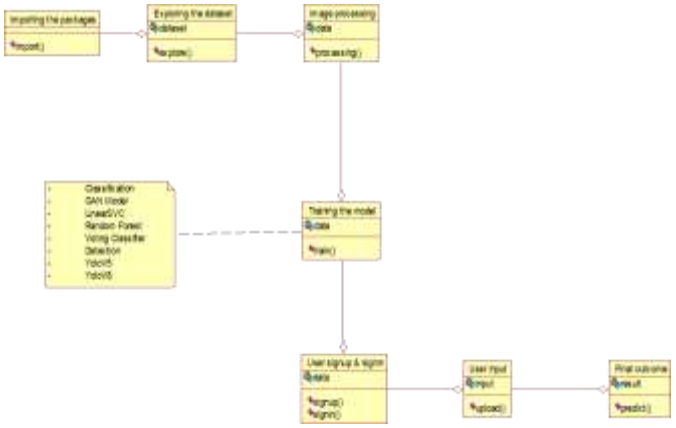


Figure 7: class diagram

ACTIVITY DIAGRAM

The activity diagram depicts the orderly progression of process flows within the system. An activity diagram is similar to a state diagram since it includes activities, actions, transitions, initial and final states, and guard conditions.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 9, No.2, September : 2024

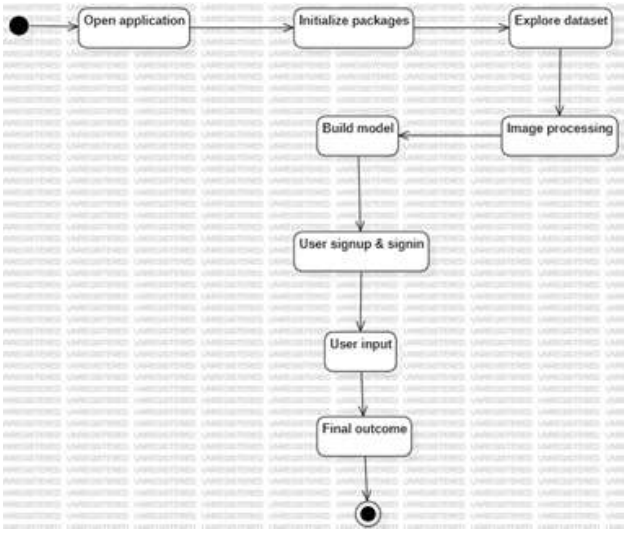


Figure 8: activity diagram

Collaboration Diagram

A collaboration diagram arranges the interactions among separate entities. The experiences are organized in a sequential manner to ease their monitoring. The cooperation diagram facilitates the discernment of all potential associations between each entity and other entities.

1: Importing libraries
2: Analyzing the dataset
3: Image processing
4: Model Construction -Classification -Generative Adversarial Network (GAN) Model -Linear Support Vector Classifier (LinearSVC) -Random Forest Classifier -Voting Classifier -Object Detection -YoloV5-YoloV8
5: User signup & sign in
6: User input

Figure 9: collaboration diagram

Component Diagram

The component diagram illustrates the essential elements of the system. This figure offers a comprehensive depiction of the numerous elements of the system and their interconnections. A component diagram visually depicts the specific components that have been chosen once The system's development or construction phase is complete.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 9, No.2, September : 2024

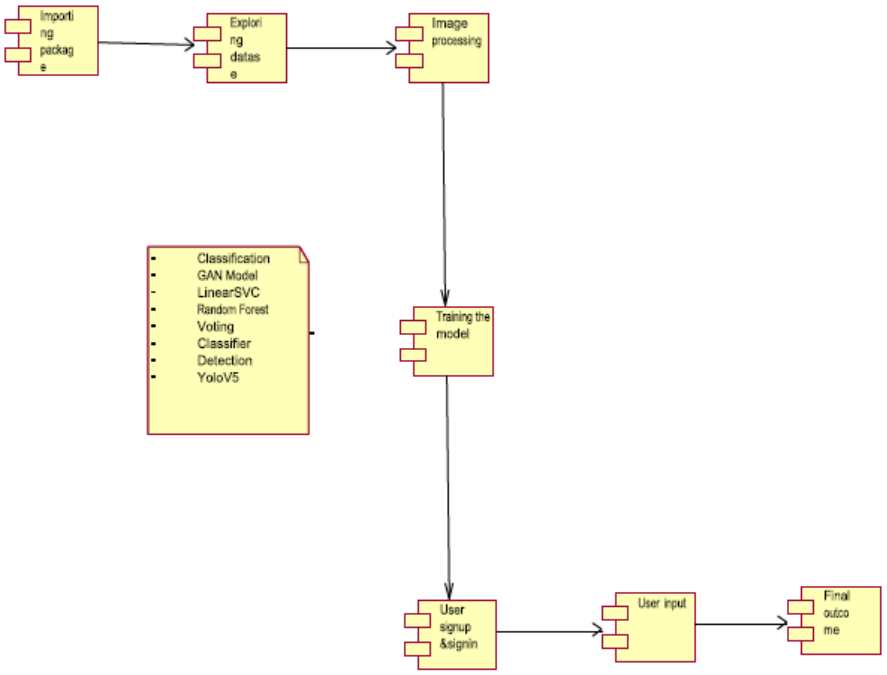


Figure 10: Flow Chart

Deployment Diagram

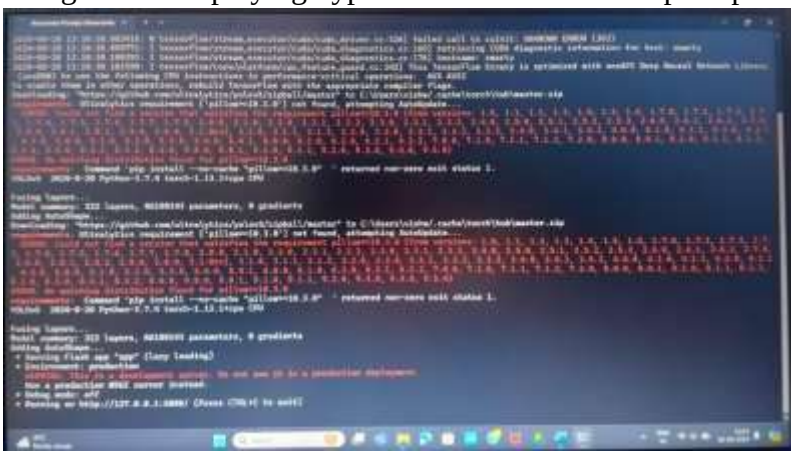
The deployment diagram illustrates the arrangement of the functional components of the application. This image is extremely beneficial once a system has been built and is ready for implementation.

Figure 11: deployment diagram

Simulation Results

For this Project, we using Anaconda Prompt (Anaconda3). The Anaconda Prompt is a terminal included in the Anaconda distribution, which functions as a command prompt. The command-line interface (CLI) enables you to effectively manage your Anaconda installation, environments, and packages. It has resemblance to the Windows Command Prompt but is pre-configured for usage with Anaconda.

Figure 12: displaying hypertext link in anaconda prompt



To begin, launch the anaconda prompt and navigate to the directory where our sample test data is located by typing "cd" followed by the file location. Press enter. Next, enter "python app.py" to execute the program. A hyperlink will appear. Copy the URL and open it in any web browser.

Click on sign up and register as a new user and sign in

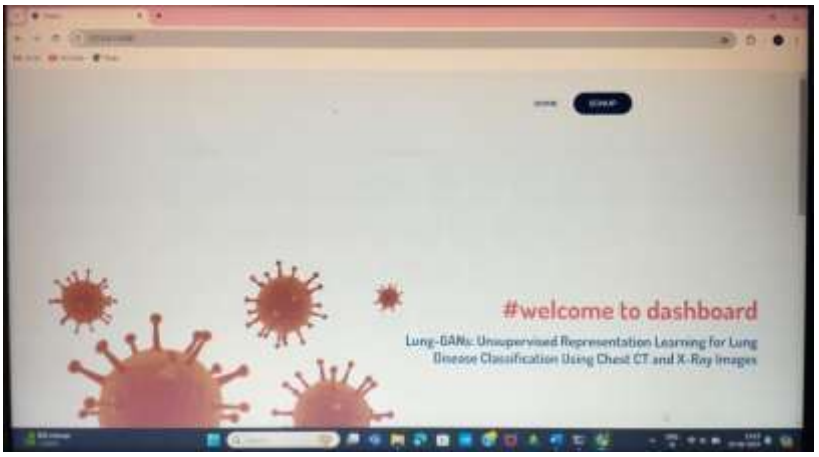


Figure 13: dashboard

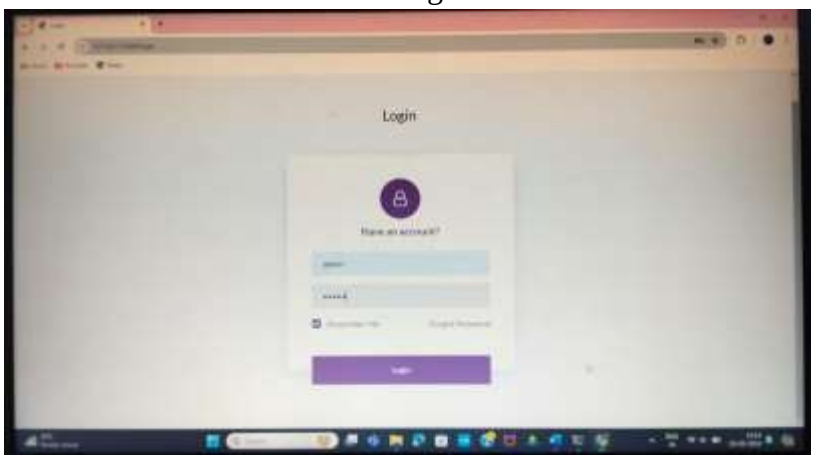


Figure 14: signup or login

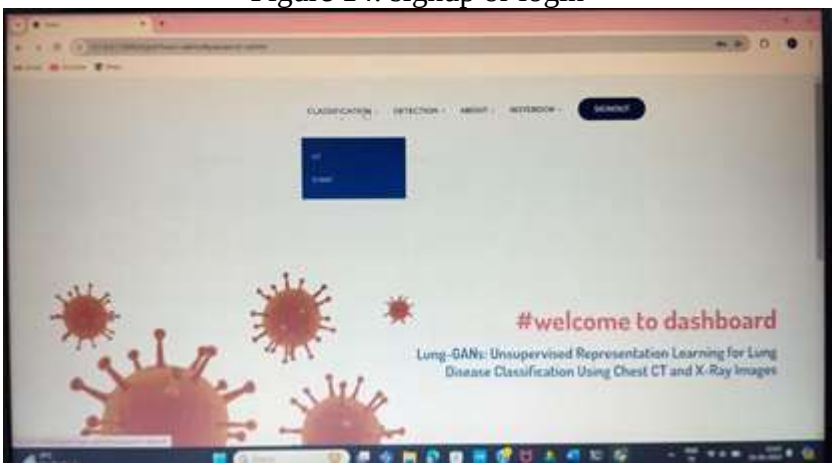


Figure 15: displaying of classification and detection

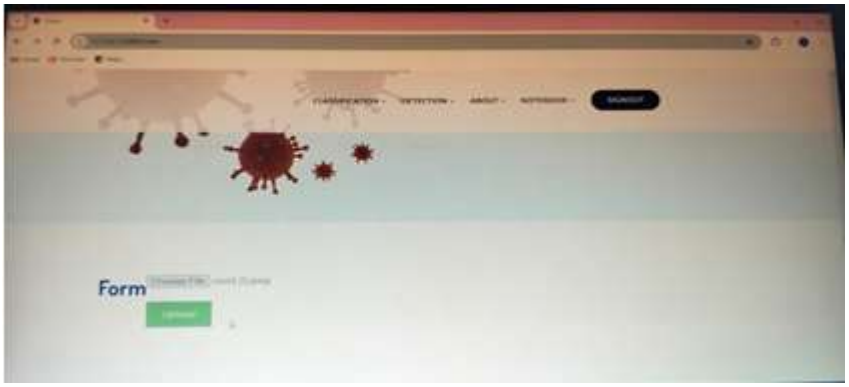


Figure 16: choosing file and uploading for classification or detection

Choose CT from the classification options and select an example picture in JPEG format. Then, proceed to upload the image.

Output Results

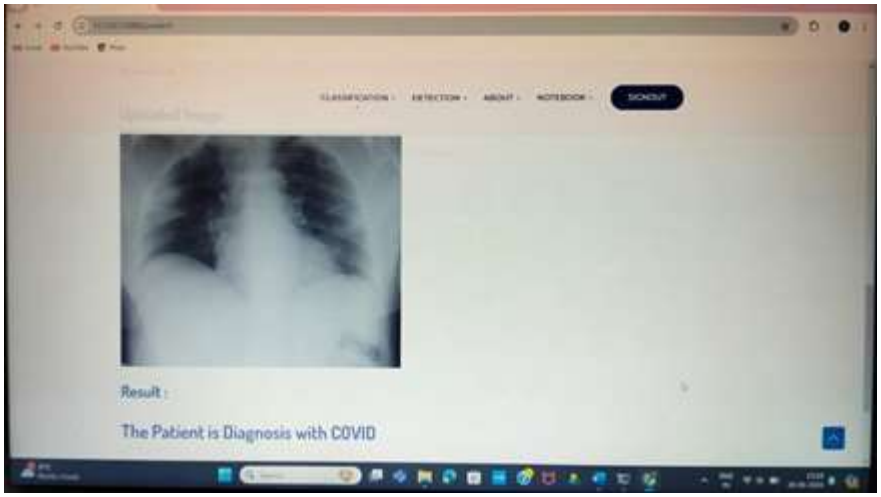


Figure 17: Output of x-ray uploaded image

Figure 17 explains that patient is dragonized with covid from the input given of x-ray

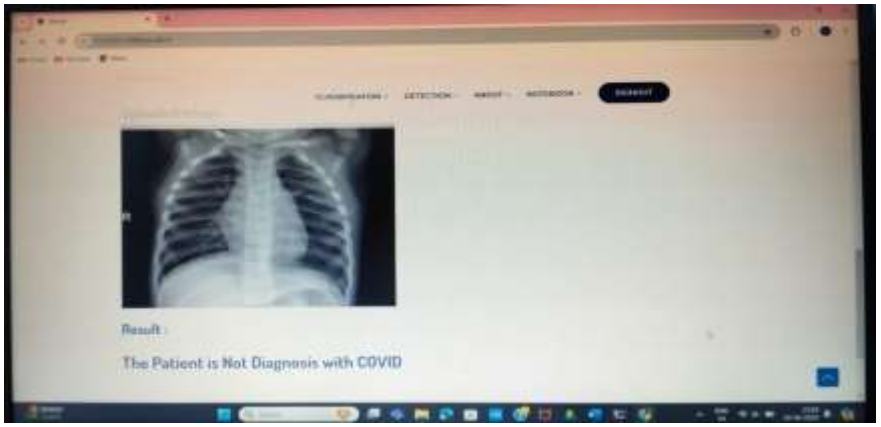


Figure 18: output of Xray uploaded image

Figure 18 shows the patient is not diagnosis with covid from the given input imageof the x-ray.

Industrial Engineering Journal
ISSN: 0970-2555
Volume : 53, Issue 9, No.2, September : 2024

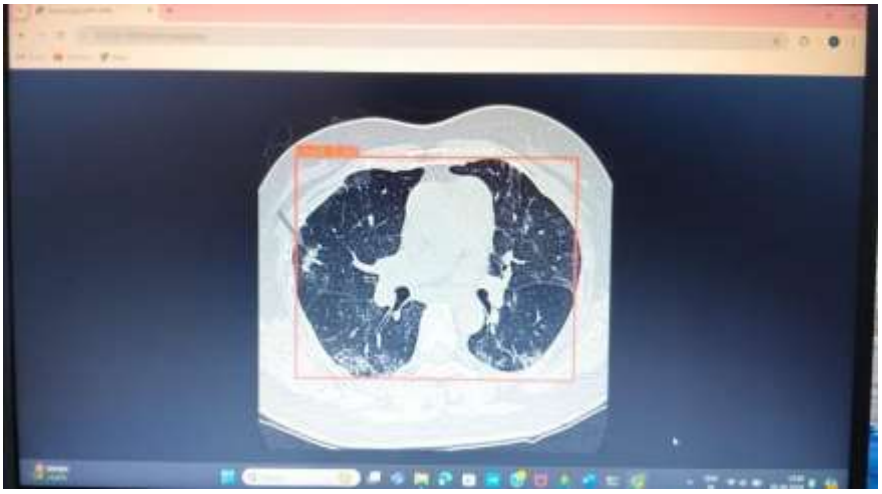


Figure 19: detection of CT uploaded image

Figure 19 Tells that, we uploaded x-ray image for detection. The output detected ascovid present in the input image and shows the area of covid present.

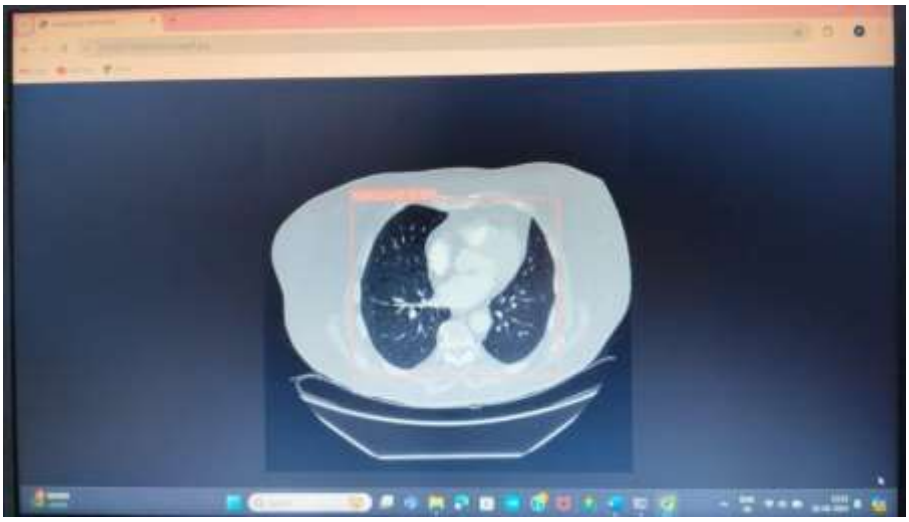


Figure 20: detection of CT uploaded image Figure 20 showsthat detection of CT uploaded image is Non covid

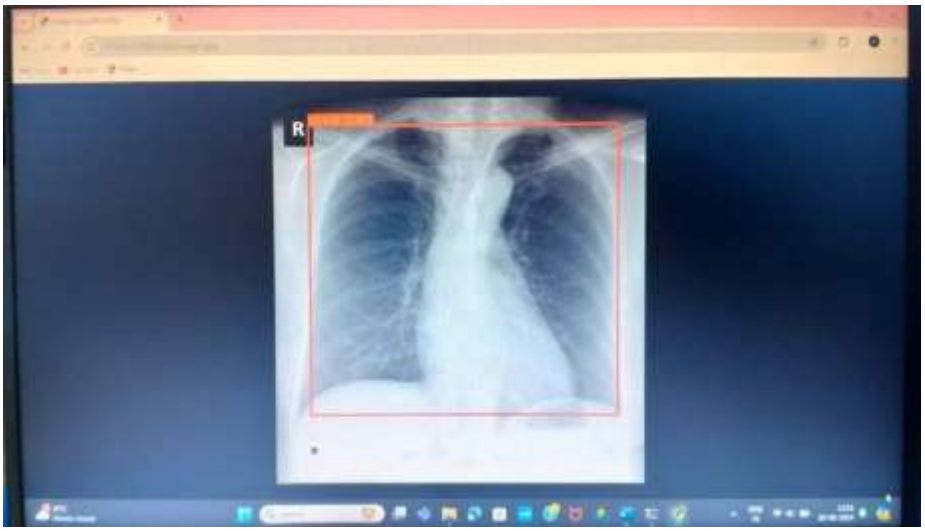


Figure 21: detection of x-ray uploaded image

Figure 21 discusses that the uploaded image of the x-ray is dragonized with covid

Figure 22: detection of x-ray uploaded image

Figure 22 shows that an uploaded input image of x-ray is not dragonizedwith covid

Conclusion

This article presents Lung-GANs, a specialized deep learning framework developed for the unsupervised classification of lung illnesses utilizing chest X-ray and CT images. The proposed methodology employs a multi-layer GAN framework to obtain visual representations of various lung diseases, such as pneumonia, tuberculosis (TB), and COVID-19. These representations are later employed in a stacking classifier that combines random forest and linear SVC to classify lung diseases. Lung-GANs has the benefit of enhancing generalization without requiring a substantial quantity of labeled data. Moreover, they have the potential to extract significant visual depictions from intricate X-ray and CT data. Moreover, the framework can be universally applied to all lung disorders without any restrictions and has the potential to be expanded to encompass other comparable respiratory issues. The model exhibited exceptional aptitude in distinguishing persons with tuberculosis from those without infection, individuals with pneumonia from those unaffected, COVID-19 patients from pneumonia patients, and COVID-19 cases from non-COVID-19 cases. Furthermore, it exhibited exceptional accuracy in all six distinct datasets pertaining to lung ailments.

References

[1] T. Lallukka, A. Millear, A. Pain, M. Cortinovis, and G. Giussani conducted a study on mortality and causes of death in 2015, in collaboration with the GBD 2015 mortality and causes of death team. The article titled "Global, regional, and national life expectancy, all-cause mortality, and cause-specific mortality for 249 causes of death, 1980-2015: A systematic analysis for the global burden of disease study 2015" was published in the Lancet journal, volume 389, issue 10064, page E1, in 2017.

[2] In his presidential presentation at the thirty-fourth annual conference of the Radiological Society of North America, L. H. Garland discusses the scientific evaluation of diagnostic procedures. This address was published in the journal Radiology, volume 52, issue 3, in 1949, spanning pages 309-328.

[3] The authors of this study, V. Ravi, H. Narasimhan, C. Chakraborty, and T. D. Pham, developed a meta-classifier strategy for COVID-19 classification utilizing CT scan and chest X-ray pictures. They utilized deep learning techniques to train their model. The study was published in the journal Multimedia Systems in 2021 and can be accessed with the DOI: 10.1007/s00530-021-00826-1.

[4] In their paper titled "Deep learning in medical image analysis," D. Shen, G. Wu, and H.-I. Suk discuss the application of deep learning techniques in the field of medical image analysis. The paper was

published in the Annual Review of Biomedical Engineering in 2017 and provides a comprehensive overview of the topic, covering all aspects and advancements in this area.

[5] The authors of this paper, A. Arunachalam, V. Ravi, V. Acharya, and T. D. Pham, have developed a platform for autonomous machine-generated data labeling and annotation. The platform is designed to be agnostic to the data model and has been applied to the specific use case of COVID-19 autoannotation. The paper is published in the IEEE Transactions on Engineering Management and spans pages 1-12. The paper can be accessed with the DOI: 10.1109/TEM.2021.3094544.

[6] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, published a paper titled "Generative adversarial nets" in the journal Advances in Neural Information Processing Systems in 2014.

[7] In December 2001, B. Van Ginneken, B. T. H. Romeny, and M. A. Viergever conducted a survey on computer-aided diagnosis in chest radiography. The results of their study were published in the IEEE Transactions on Medical Imaging, volume 20, issue 12, pages 1228-1241.

The paper titled "Deep convolutional neural networks for chest diseases detection" by R. H. Abiyev and M. K. S. Ma'aitah was published in the Journal of Healthcare Engineering in 2018. The paper explores the use of deep convolutional neural networks for detecting chest diseases.

[9] The paper titled "Automated abnormality classification of chest radiographs using deep convolutional neural networks" was authored by Y.-X. Tang et al. It was published in NPJ Digital Medicine in 2020, in volume 3, issue 1, pages 1-8.

[10] In their paper titled "Deep-learning framework to detect lung abnormality: A study with chest X-ray and lung CT scan images," Bhandary et al. present a pattern recognition approach that utilizes deep learning to identify abnormalities in the lungs. The study was published in the journal Pattern Recognition Letters in 2020, and it provides detailed insights into the framework's performance and effectiveness.

[11] S. Bharati, P. Podder, and M. R. H. Mondal conducted a study titled "Hybrid deep learning for detecting lung diseases from x-ray images" which was published in the journal Inform. Med. Unlocked in 2020. The article is identified by the article number 100391.

[12] In 2020, J. Ma, Y. Song, X. Tian, Y. Hua, R. Zhang, and J. Wu conducted a survey on the use of deep learning in pulmonary medical imaging. The survey was published in the journal Frontiers in Medicine, volume 14, issue 4, and the pages 450-469.

[13] In their paper titled "A survey of deep learning for lung disease detection on medical images: State-of-the-art, taxonomy, issues and future directions," S. T. H. Kieu, A. Bade, M. H. A. Hijazi, and H. Kolivand provide an overview of the current advancements in deep learning techniques for the detection of lung diseases in medical images. The authors discuss the existing state-of-the-art methods, present a taxonomy of these techniques, highlight the challenges and issues faced in this field, and propose future directions for research. This paper was published in the Journal of Imaging, volume 6, issue 12, page 131, in the year 2020.

[14] The authors of this paper, A. A. Saraiva, N. M. F. Ferreira, L. L. de Sousa, N. J. C. Costa, J. V. M. Sousa, D. Santos, A. Valente, and S. Soares, developed a method for classifying pictures of juvenile pneumonia using convolutional neural networks. This research was published in the journal BIOIMAGING in 2019, on pages 112-119. The paper may be accessed with the DOI: 10.5220/0007404301120119.

[15] The paper titled "A novel transfer learning-based approach for pneumonia detection in chest X-ray images" was published in the journal Applied Science in 2020. It was authored by V. Chouhan et al. The paper is focused on proposing a new method for detecting pneumonia in chest X-ray images using transfer learning.