KHAZAR UNIVERSITY

School: Graduate School of Science, Art and Technology

Department: Computer Science

Specialty: 60631 - Computer Engineering

MASTER'S THESIS

Theme: Prediction of COVID-19 using procedures of transfer learning

Student: Sevinj Aliyeva Ahsan

Supervisor: Prof. Dr. Shahnaz Shahbazova

Baku – 2024

XƏZƏR UNİVERSİTETİ

Fakültə: Təbiət elmləri, Sənət və Texnologiya Yüksək Təhsil

Departament: Kompüter elmləri

İxtisas: 60631 – Kompüter mühəndisliyi

MAGİSTR TEZİSİ

Mövzu: Transfer təlim prosedurlarından istifadə edərək COVID-19-un proqnozlaşdırılması

Magistrant: Sevinc Əhsən qızı Əliyeva

Elmi rəhbər: t.e.d., prof. Şahnaz Şahbazova

Bakı - 2024

TABLE OF CONTENTS

INTRODUCTION

The emergence of the COVID-19 pandemic, originating from the novel coronavirus SARS-CoV-2 in late 2019 in Wuhan, China, led to unprecedented global challenges for healthcare systems. Its profound impact affected various aspects of life, including public health, the economy, social interactions, and mental well-being. Healthcare systems worldwide faced significant strain as they struggled to cope with the sudden surge in patients requiring medical care. Hospitals became overwhelmed, resulting in shortages of essential medical supplies such as personal protective equipment (PPE), ventilators, and hospital beds. Healthcare workers endured immense pressure and exhaustion from the demanding task of caring for COVID-19 patients. Moreover, the pandemic exacerbated existing healthcare disparities, disproportionately affecting vulnerable populations such as the elderly, racial and ethnic minorities, and individuals with underlying health conditions. Factors such as limited access to healthcare services, socioeconomic issues, and systemic inequalities contributed to variations in infection rates and outcomes. The early and accurate detection of COVID-19 cases played a crucial role in managing and containing the disease. Rapid identification of cases facilitated the prompt implementation of isolation measures, contact tracing, and quarantine protocols, which are vital for slowing the spread of the virus. Additionally, early detection enabled timely medical intervention and supportive care for individuals with severe symptoms, potentially reducing morbidity and mortality rates. Various diagnostic tests, including molecular (PCR) tests, antigen tests, and antibody tests, were developed to detect COVID-19 infections. These tests served important roles in screening individuals, diagnosing active infections, and identifying past exposure to the virus. However, challenges such as test availability, accuracy, and turnaround times hindered widespread testing efforts.

The significance of utilizing transfer learning for COVID-19 detection stems from various factors:

- Limited Data Availability: During the early stages of the pandemic, there was a scarcity of labeled COVID-19 datasets for training machine learning models. Transfer learning allows for the adaptation and fine-tuning of pre-trained models from related tasks, such as medical imaging or general pathology, even with limited COVID-19-specific data.

- Speed and Efficiency: Transfer learning expedites the development of COVID-19 detection models by leveraging pre-existing knowledge encoded in pre-trained models. This minimizes the necessity for extensive data collection and model training from scratch, facilitating quicker deployment of diagnostic tools to combat the pandemic.
- Generalizability: Transfer learning enables the creation of robust and adaptable COVID-19 detection models that perform effectively across various populations, healthcare environments, and imaging modalities. By transferring knowledge gleaned from diverse datasets, these models can adjust to differences in image quality, patient demographics, and disease presentations.
- Resource Conservation: Developing machine learning models from the ground up demands substantial computational resources and expertise. Transfer learning conserves resources by repurposing pre-trained models and fine-tuning them on smaller COVID-19 datasets, making it more accessible to researchers and healthcare professionals with limited resources (Lahsaini et al., 2021).

In contrast, traditional diagnostic methods for COVID-19, such as RT-PCR tests and imaging techniques like chest X-rays and CT scans, pose several limitations:

- Labor-intensive and Time-consuming: RT-PCR tests, the gold standard for diagnosing COVID-19, require specialized equipment, reagents, and trained personnel to administer. Additionally, obtaining results can be time-consuming, potentially delaying patient care and public health interventions.
- Subject to False Negatives and False Positives: RT-PCR tests may yield false-negative results, especially in cases with low viral loads or improper sample collection. Conversely, imaging techniques like chest X-rays and CT scans may produce false-positive results, leading to unnecessary interventions and resource utilization.
- Dependence on Human Interpretation: The interpretation of diagnostic tests such as chest X-rays and CT scans relies on the expertise of radiologists, introducing subjectivity and variability in diagnosis. Moreover, interpretation may be influenced by factors like fatigue and experience, affecting diagnostic accuracy (Jia et al., 2021).

Artificial intelligence (AI) and machine learning (ML) techniques offer promising solutions to address these limitations and enhance diagnostic accuracy for COVID-19 detection:

- Automated Analysis: AI and ML algorithms can automate the analysis of diagnostic tests, reducing reliance on manual interpretation and potentially enhancing consistency and efficiency.
- Enhanced Sensitivity and Specificity: ML models can be trained to identify subtle patterns and features indicative of COVID-19 infection, potentially improving the sensitivity and specificity of diagnostic tests compared to traditional methods.
- Integration of Multimodal Data: AI techniques enable the integration of diverse data sources, such as clinical data, imaging studies, and laboratory results, to develop comprehensive diagnostic models that leverage complementary information for improved accuracy.
- Real-time Decision Support: AI-powered diagnostic tools can provide real-time decision support to healthcare providers, assisting in swift and accurate diagnosis, triage, and patient management, particularly in resource-constrained settings or during surges in COVID-19 cases (Li et al., 2024). Table 1 presents a comparison between Transfer Learning and AI/ML Techniques versus Traditional Diagnostic Methods for COVID-19 Detection.

Table 1. Comparison of Transfer Learning and AI/ML Techniques versus Traditional Diagnostic Methods for COVID-19 Detection

Aspect	Transfer Learning and AI/ML Techniques	Traditional Diagnostic Methods
Data Availability	Utilizes pre-trained models and adapts them even with limited COVID-19 data	Dependent on availability of labeled datasets
Speed and Efficiency	Speeds up model development by leveraging pre-existing knowledge	Labor-intensive and time-consuming procedures
Generalizability	Creates robust models effective across populations and imaging modalities	Subject to variability in interpretation and diagnostic accuracy

Aspect	Transfer Learning and AI/ML Techniques	Traditional Diagnostic Methods
Resource Conservation	Conserves computational resources and expertise	Requires specialized equipment and trained personnel
Automated Analysis	Automates diagnostic test analysis, reducing manual interpretation	Relies on human interpretation, introducing subjectivity
Enhanced Sensitivity/Specificity	Identifies subtle patterns for improved sensitivity and specificity	Prone to false negatives and false positives
Integration of Multimodal Data	Integrates diverse data sources for comprehensive diagnostic models	Limited to individual test modalities
Real-time Decision Support	Provides real-time decision support for swift and accurate diagnosis	May lead to delays in patient care due to manual processes

The dataset utilized in this research was sourced from Kaggle and encompasses chest X-ray images. Initially, the dataset exhibited imbalances, with varying quantities of images across the different classes. To rectify this, the authors employed a combination of under-sampling and over-sampling techniques to ensure a harmonized representation of each class. A novel Singular Value Decomposition (SVD)-based image processing method was employed by the authors to augment the minor classes, specifically Viral Pneumonia and COVID. This approach introduced subtle alterations in luminance and contrast, thereby enriching the diversity of images within these classes. Furthermore, Contrast Limited Adaptive Histogram Equalization (CLAHE) was applied to the entire dataset to accentuate features in the chest X-ray images. The empirical selection of CLAHE parameters was meticulously executed to achieve optimal contrast enhancement without introducing excessive artifacts**.** It is pertinent to highlight that, for training purposes, a subset of images was chosen from each class, resulting in a balanced dataset with approximately equivalent representation across all classes. Nevertheless, the count of images per class in the augmented dataset may exhibit slight disparities, reflecting the inherent variability within each class. The authors conducted a comprehensive analysis of intra-class variance utilizing correlation coefficients to justify the selection of images for the augmented dataset. The dataset encompasses both training and testing images, with the testing images provided in a distinct folder devoid of preprocessing. This partitioning facilitates an unbiased evaluation of model performance on unseen data, ensuring the robustness and generalizability of the developed models.

The main goal of this research is to create and assess a model based on transfer learning for the automatic identification of COVID-19 using chest X-ray images. The specific objectives are:

- Assessing the effectiveness of transfer learning techniques in applying pre-trained convolutional neural networks (CNNs) for COVID-19 detection.
- Analyzing the model’s ability to accurately differentiate between COVID-19 cases and other respiratory conditions such as pneumonia and lung opacity.
- Testing the model’s robustness and generalization across different datasets and imaging methods.
- Comparing the proposed model’s performance with existing methods for COVID-19 detection using medical imaging data.

Providing insights into the potential clinical applications and implications of the model to help healthcare professionals diagnose COVID-19 cases more efficiently and accurately.The study focuses on developing and evaluating a transfer learning-based approach for COVID-19 detection using chest X-ray images. The scope of the research includes the following components:

- Dataset: Utilizing a publicly available dataset from Kaggle, containing chest X-ray images classified into four categories: COVID, Lung opacity, Normal, and Viral Pneumonia. Balancing of the dataset is ensured through under-sampling and over-sampling techniques to maintain equal representation of each class.
- Methodology: Employing transfer learning techniques to leverage pre-trained CNN models for feature extraction and classification. Data preprocessing, model training, and evaluation processes are carried out using Python programming language and popular deep learning frameworks like TensorFlow.
- Evaluation Metrics: Assessing the performance of the developed model using standard evaluation metrics such as accuracy, precision, recall, F1-score and confusion matrix. Additionally, qualitative analysis of model predictions and visualizations may be conducted to gain insights into the model's behavior and decision-making processes.

- Limitations: Acknowledging various limitations and constraints associated with the study, including dataset availability and quality, computational resources, and potential biases in model predictions. Furthermore, the study focuses solely on chest X-ray images for COVID-19 detection and does not consider other imaging modalities or clinical data.
- Ethical Considerations: Adhering to ethical guidelines and regulations governing the use of medical data and machine learning techniques in healthcare research. Measures are taken to ensure patient privacy, data security, and transparency in reporting research findings (Roy et al., 2022).

The thesis is structured into several parts: Introduction, Literature Review, Methodology, Implementation and training, Results and discussion, and Conclusion.

This research aims to address challenges in early COVID-19 detection by applying transfer learning to chest X-ray images. Its significance lies in contributing to COVID-19 diagnosis efforts, improving medical imaging and healthcare technology, promoting transparency and collaboration in research, and potentially influencing future healthcare solutions beyond the pandemic.

In conducting this research, ethical considerations are paramount. I recognize the sensitive nature of medical data, particularly patient images, and the importance of maintaining patient privacy and confidentiality. Precautions have been taken to de-identify and anonymize all data used in this study in order to protect the privacy of individuals. Additionally, I will make efforts to mitigate any biases in the dataset and analysis to ensure fair and equitable results. Furthermore, transparency will be maintained throughout the research process, and all findings will be reported accurately and objectively to contribute to the advancement of knowledge in the field of medical imaging and COVID-19 diagnosis.

In conclusion, this introduction has provided an overview of the COVID-19 pandemic, the dataset sourced for this study, and the objectives of the research. By leveraging transfer learning techniques on medical imaging data, I aimed to develop a model for automated detection of COVID-19, contributing to the global efforts in combating the pandemic. The subsequent chapters will delve into the methodology, results, and discussions, offering insights into the effectiveness of the proposed approach and its implications for healthcare. Through this research, I aspired to enhance diagnostic capabilities and ultimately improve patient outcomes in the fight against COVID-19.

Research Questions:

- What challenges arise in using pre-trained models for COVID-19 prediction?
- How do the developed models perform compared to existing approaches?

Relevance:

Accurate prediction methods for COVID-19 are crucial in combating its global impact. Utilizing advanced machine learning, particularly transfer learning, addresses the pressing need for rapid and reliable diagnostic tools in healthcare.

Importance:

Effective diagnostic methods are urgently needed to manage the spread of COVID-19. Leveraging transfer learning enhances the accuracy of COVID-19 detection from chest X-ray images, impacting patient outcomes, healthcare resource optimization, and public health strategies. This research also advances medical imaging and machine learning fields.

Object:

The research aims to develop and evaluate machine learning models for COVID-19 prediction from chest X-ray images. Specifically, it focuses on fine-tuning transfer learning techniques to adapt models to the COVID-19 dataset's unique characteristics.

Subject:

Various machine learning architectures, such as VGG16, EfficientNet, InceptionV3, and MobileNet, are investigated in classifying chest X-ray images as COVID-19 positive or negative. The study provides a detailed comparative analysis to determine the most effective model for this task.

CHAPTER 1. LITERATURE REVIEW

During the ongoing COVID-19 pandemic, there has been a notable upsurge in interest in harnessing sophisticated computational methods, notably transfer learning, for forecasting and analyzing the spread of the disease and its repercussions. This literature review aims to investigate and amalgamate significant contributions and advancements in the realm of COVID-19 prediction employing transfer learning methodologies. The adoption of transfer learning in the context of COVID-19 forecasting signifies a considerable departure from conventional epidemiological methods, presenting promising avenues for enhancing prediction accuracy and decision-making procedures. Through the utilization of pre-existing models and insights from related fields, transfer learning empowers researchers to tailor predictive models specifically for COVID-19 dynamics, potentially augmenting the efficacy of public health interventions and resource allocation strategies.

This review will scrutinize seminal works and recent research endeavors in the arena of transfer learning applied to COVID-19 prediction, delving into the methodologies employed, data sources utilized, and the performance metrics evaluated across various studies. By offering insights into the strengths, limitations, and prospective directions of transfer learning-based approaches in combating the pandemic, it aims to contribute to the evolving body of knowledge in this critical area.

Marios Constantinou et al. (2023) aimed to utilize advanced machine learning methods, particularly transfer learning, for the automated identification of COVID-19 in chest X-ray images. The main dataset used is the COVID-QU dataset, which comprises 33,920 CXR images categorized into three classes: COVID-19, non-COVID-19, and Normal. To compile this dataset, various sources, including databases, repositories, and medical schools, were accessed. The dataset was then divided into training, validation, and test sets, ensuring balanced representation across the classes in each subset. Five cutting-edge Convolutional Neural Network (CNN) architectures, namely ResNet50, ResNet101, DenseNet121, DenseNet169, and InceptionV3, were assessed for their ability to classify COVID-19 from CXR images. These models underwent pre-training on the ImageNet dataset and fine-tuning on the COVID-QU dataset. Techniques such as random rotation and horizontal flip were employed to augment the data and mitigate overfitting. Image resizing was performed to accommodate the specific requirements of each architecture. The performance of each model was evaluated using precision, recall, and F1-score metrics, with a particular emphasis on achieving high recall rates for COVID-19 detection. ResNet101 emerged as the most effective model, achieving a recall of 96% across all metrics. However, it had the highest number of trainable parameters, indicating greater computational complexity compared to the other models. In conclusion, this study underscores the effectiveness of transfer learning in COVID-19 detection from chest X-ray images and emphasizes the significance of high recall in identifying COVID-19 positive cases.

Salih Sarp et al. (2023) aimed to improve the automated identification of COVID-19 in chest X-ray images by integrating lung segmentation and utilizing transfer learning methods. Lung segmentation is introduced to enhance the detection and interpretation processes by emphasizing features within the lung area and facilitating explanations within the framework. Transfer learning is then employed to accelerate feature extraction and categorization, especially advantageous given the limited availability of X-ray images. Following feature extraction, the X-rays undergo classification, and the model's efficacy is assessed. A crucial component of the approach involves highlighting regions suggestive of COVID-19 pneumonia using a LIME-based heatmap explanation, aiding physicians in non-invasive diagnosis. In the lung segmentation phase, a hybrid U-Net architecture is deployed, incorporating a pre-trained VGG11 feature extractor. Diverse augmentation techniques are applied, achieving high Jaccard and dice scores. Transfer learning entails utilizing pre-trained deep learning models such as VGG-Net, ResNet, and Inception V3 to enhance feature extraction and classification, adapted for COVID-19 detection tasks based on large datasets like ImageNet. Explainable artificial intelligence (XAI) methods like LIME are employed to enhance the interpretability of AI models. LIME generates heatmap explanations to pinpoint areas indicative of COVID-19 pneumonia in X-ray images, assisting in diagnosis and prognosis. The performance of the proposed model is compared against models lacking transfer learning and lung segmentation, demonstrating enhanced resource management and efficiency. The study illustrates the potential of AI, coupled with XAI tools, for expedited and more precise diagnosis and monitoring of COVID-19, particularly when analyzing chest X-ray images.

Arora et al. (2021) in this study aimed to identify whether lung CT scans indicate the presence or absence of COVID-19. This process involves using a residual dense neural network (RDNN) to preprocess CT images, improving spatial resolution while keeping costs and complexity low. RDNN overcomes the drawbacks of traditional super-resolution methods by utilizing deep learning algorithms to extract detailed information from CT scans. Two standard datasets, SARS-COV-2 CT and COVID-CT were utilized for the transfer learning models in this research. These datasets were split into training and testing sets, with the COVID-CT-Dataset comprising 349 COVID-19 CT images and 463 non-COVID-19 CTs, while the SARS-COV-2 CT included 2482 CT scan images. The RDNN method involved employing a residual dense network (RDN) to tackle issues with low-resolution in medical imaging. RDN uses residual dense blocks (RDB) for feature extraction, hierarchical feature fusion, and up-sampling techniques to enhance CT image resolution. Additionally, image augmentation methods were used to diversify the dataset and enhance classification accuracy. Geometric adjustments were applied to augment lung CT images, increasing the dataset's size. Transfer learning models like DenseNet121, MobileNet, VGG16, ResNet50, InceptionV3, and XceptionNet were utilized for classification. These models were pretrained on the ImageNet dataset and adapted for COVID-19 detection. Performance evaluation metrics such as accuracy, precision, recall, and F1 score were employed to gauge model effectiveness. Results showed enhanced performance when employing super-resolution techniques, indicating the effectiveness of the proposed methodology for identifying COVID-19 in lung CT scans.

This research conducted by Sai Zhang and Guo-Chang Yuan (2022) focused on creating deep learning models to automatically detect COVID-19 using chest CT scans, employing convolutional neural network (CNN) architectures known for their effectiveness in image classification. Specifically, the study examines the performance of VGG19 and ResNet50V2 architectures, with the goal of improving their efficiency in COVID-19 diagnosis. To enhance model performance, they introduce a new method involving a 2D global max pooling layer instead of traditional layers like flatten or 2D global average pooling. For VGG19-based models, they utilize transfer learning by initializing the pretrained convolutional layers with ImageNet weights. These models include additional layers such as a flatten layer, dense layer, dropout layer, and densely connected classifier. Similarly, ResNet50V2-based models employ the residual learning framework and pretrained convolutional layers for feature extraction. These models are then compared with other advanced deep learning architectures, including Vision Transformer, MobileNetV2, InceptionResNetV2, and ResNet152V2. The training process involves data preprocessing steps such as normalization and augmentation, where augmentation techniques are applied to diversify the dataset and reduce overfitting. Each model undergoes training using a binary cross-entropy loss function and the Adam optimizer, with specific layers frozen and jointly trained to optimize performance. The results show high training and validation accuracies for both VGG19 and ResNet50V2-based models, surpassing benchmarking models. Notably, the use of the 2D global max pooling layer improves COVID-19 detection accuracy by around 1%. Their top-performing model, VGG19 with 2D global max pooling, achieves an accuracy of 94.12%, sensitivity of 91.40%, specificity of 96.95%, false discovery rate (FDR) of 3.11%, and an area under the curve (AUC) of 0.9744. Additionally, they introduced a heatmap method to highlight lesion areas in COVID-19 chest CT images, aiding in abnormal pattern identification. They also develop online simulation software for COVID-19 detection using CT images, enabling rapid radiology checks with a classification speed as fast as 1.1 ms per CT image. Ensuring ethical approval, the current clinical trial follows the principles of the Declaration of Helsinki and the International Conference on Harmonization–Good Clinical Practice guidelines.

The work of Arpita Halder and Bimal Datta (2021) on identifying COVID-19 using lung CT scans, utilizing a dataset compiled by Angelov et al. This dataset consists of genuine patient CT scans gathered from hospitals in Sao Paulo, Brazil, and is accessible on Kaggle. It contains 2481 CT scan images categorized into COVID and non-COVID groups, with 1252 scans from COVID-positive patients and 1229 from COVID-negative patients. An 8:2 ratio was used for the split between training and testing data. The methodology involves developing a two-dimensional DL framework named KarNet, which relies on transfer learning. Transfer learning, renowned for constructing models swiftly with limited datasets, was utilized in three stages: data pre-processing, feature extraction, and binary classification. Four pre-trained models (DenseNet201, MobileNet, ResNet50V2, and VGG16) were employed, each supplemented with extra layers to assess performance on augmented and unaugmented datasets. CT-scan images underwent pre-processing and normalization before being inputted into the pre-trained models for feature extraction. The models' convolutional bases were re-trained, and classifiers were substituted for binary classification. Data augmentation methods, such as image rotation, shifting, and horizontal flipping, were employed to prevent overfitting. KarNet surpassed previous models, achieving remarkable accuracy, with DenseNet201 exhibiting the best performance. Tensorflow was utilized for implementation, allowing efficient utilization of computational resources. The KarNet model exhibited exceptional diagnostic performance, achieving 96.79% accuracy on both augmented and unaugmented datasets.

Other models (VGG16, ResNet50V2, MobileNet) also demonstrated promising accuracy. The proposed framework enhances diagnostic capabilities and could be integrated into hospitals with CT scanners, offering an automated option for COVID-19 testing, potentially saving time and lives.

The methodology adopted by Sahil Lawton and Serestina Viriri (2021) adheres to a standard image processing sequence involving data acquisition, preprocessing, segmentation, feature extraction, and classification. Initially, the original dataset, alongside two additional duplicates, undergoes Histogram Equalization (HE) and Contrast Limited Adaptive Histogram Equalization (CLAHE), yielding three distinct datasets. Subsequently, each dataset is partitioned into training, validation, and testing subsets in a 60:20:20 distribution, with data augmentation applied to the training data to prevent overfitting. Automatic segmentation and feature extraction are executed by the convolutional bases of transfer learning models, while classification is carried out using a fully connected artificial neural network. The dataset utilized, known as the SARS-CoV-2 CT scan dataset, comprises 2482 images sourced from patients in São Paulo, Brazil, categorized into COVID-19-positive and COVID-19-negative classes. Histogram Equalization (HE) and Contrast Limited Adaptive Histogram Equalization (CLAHE) techniques are deployed to amplify image contrast and enhance feature extraction. Data augmentation strategies are employed to augment the scale and quality of the training dataset, thereby enhancing the performance and adaptability of the deep learning model. Experiments are conducted utilizing five transfer learning architectures—ResNet-101, VGG-19, DenseNet201, EfficientNet-B4, and MobileNet-V2—augmented with a fully connected hidden layer and softmax output layer. Training entails iterative updates of convolutional base and classifier weights using the categorical cross entropy loss function and RMSprop optimizer across 200 epochs. Performance evaluation metrics encompassing accuracy, precision, recall, F1-score, specificity, and ROC-AUC are employed to gauge the models' efficacy on the testing dataset. The VGG-19 architecture coupled with Contrast Limited Adaptive Histogram Equalization (CLAHE) emerges as the top-performing model. The study underscores the potential of transfer learning models in streamlining COVID-19 detection from lung CT scans, offering an alternative approach to traditional testing methodologies. Future research avenues include delving into automatic hyperparameter optimization techniques and crafting transfer learning-based frameworks tailored for processing 3D CT scans.

The research conducted by Shubham Agrawal et al. (2023) aimed to employ advanced machine learning techniques, particularly transfer learning, to automate the identification of COVID-19 from chest X-ray images. The dataset utilized in this study, curated by Cohen et al., comprises 125 chest X-ray images diagnosed with COVID-19, along with 500 images each for no-findings and pneumonia, sourced from publicly available repositories. Various deep learning models were assessed using the mean accuracy of five-fold cross-validation, where each model underwent training on five distinct sets of 900 images and testing on 225 images. Transfer learning, a strategy involving the transfer of knowledge from pre-trained deep learning models to novel tasks, was utilized to enhance the performance of the classification model. Multiple established models including VGG, InceptionV3, ResNet, MobileNetV2, DenseNet121, and Xception were scrutinized to devise the proposed model. This proposed model, a modified version of ResNet50 based on transfer learning, surpassed other models in terms of accuracy. To ensure a comprehensive analysis, k-fold cross-validation was implemented, ensuring consistent evaluation of all models. Additionally, the dataset, code files, and output images are made accessible online to promote replicable research. Model interpretation methods such as LIME (Local Interpretable Model-Agnostic Explanations) and Grad-CAM (Gradient-weighted Class Activation Mapping) were employed to interpret the models' decisions and identify areas for enhancement. The experimental findings demonstrated that the proposed model achieved notable accuracy in distinguishing COVID-19 and no-findings, with a mean accuracy of 99.20%. In classifying COVID-19, no COVID-19, and pneumonia, the proposed model exhibited an accuracy of 86.13%. A detailed analysis encompassing metrics like accuracy, recall, specificity, precision, and F1-score for both scenarios underscores the effectiveness of the proposed model in detecting COVID-19 from chest X-ray images.

The introductory segment of the research by Linh T. Duong et al. (2023) delves into the fundamental concepts essential for grasping their methodology. They commence by introducing two families of deep neural networks, namely EfficientNet and MixNet, which serve as the foundational framework for the classification process in their study. While EfficientNet aims to strike a balance between accuracy and computational efficiency through scaling in width, depth, and resolution dimensions, MixNet is geared towards reducing parameters and computational complexity, drawing inspiration from the MobileNets architecture. Both families offer diverse configurations tailored to specific scaling requirements. Transfer learning, a central tenet in their research, is succinctly discussed as a method for fine-tuning hyperparameters in deep neural networks. This approach facilitates the transfer of knowledge from a well-established source domain to a new target domain, thus streamlining model training with a relatively smaller labeled dataset. They explore three distinct learning methodologies: ImageNet, AdvProp, and NS, each presenting unique advantages in model training and enhancing accuracy. Their primary research objective is to develop an expert system utilizing EfficientNet and MixNet for automated COVID-19 detection from chest X-ray (CXR) and lung computed tomography (LCT) images. To achieve this, they propose three separate learning strategies—ImageNet, AdvProp, and NS—to expedite learning processes and enhance accuracy. Subsequently, the evaluation section elaborates on the datasets and methodologies employed to evaluate the performance of their approach. During the evaluation phase, they employ four real-world datasets alongside recent implementations of EfficientNet and MixNet. They integrate pretrained weights from various sources to expedite training and ensure model robustness. Their evaluation criteria focus on metrics such as accuracy, precision, recall, and F1 score to gauge the effectiveness of their approach in predicting COVID-19 from CXR and LCT images. Additionally, they assess the computational efficiency of their models to ensure practical applicability. Upon analyzing the results, they observed that configurations utilizing EfficientNet and specific transfer learning methodologies demonstrate superior performance in accurately detecting COVID-19 from CXR and LCT images. Their approach surpasses existing studies in terms of accuracy, precision, recall, and F1 score, thereby affirming the efficacy of EfficientNet and MixNet in COVID-19 detection. Furthermore, EfficientNet proves particularly adept at handling large-scale datasets, underscoring its potential for real-world applications.

A study conducted by Ramachandran (2021) delves into the utilization of transfer learning, a technique harnessing the knowledge from pre-trained models, particularly ResNet, for COVID-19 detection from lung CT images. Deep Convolutional Neural Networks (CNNs) rely heavily on the size of training datasets for accuracy, but training them with extensive datasets is time-consuming. Transfer learning circumvents this by leveraging pre-existing knowledge. The process involves freezing initial layers, which contain generic information, and fine-tuning subsequent layers to learn specific features from the current dataset. Two methods of transfer learning are employed: freezing initial layers and training only subsequent layers, or fixing all layers except the classification layer. ResNet, known for its performance with numerous layers, employs skip connections to mitigate the vanishing gradient problem. Three ResNet models (ResNet50, ResNet101, ResNet152) are used for classifying lung CT images, where the initial layers remain frozen, and subsequent layers are fine-tuned on COVID-19 CT lung image datasets. Fine-tuning involves unfreezing specific layers to learn features specific to COVID-19 CT images. The models are evaluated based on accuracy and loss during training and testing stages. Results show improved accuracy after fine-tuning, with ResNet152 performing the best during testing. Comparison with existing methods demonstrates the efficacy of the proposed approach, particularly with ResNet152 outperforming other methods utilizing chest X-Ray images.

Benbrahim et al. (2020) employed Deep Learning Pipelines on Apache Spark to expedite transfer learning. Their method involved utilizing pre-trained CNN architectures, specifically InceptionV3 and ResNet50, in conjunction with logistic regression to classify chest X-ray images. Deep Learning Pipelines, integrated into Databricks Runtime ML, served as the framework for their deep learning workflows within the Apache Spark environment. The dataset comprised chest X-ray images sourced from two repositories: "COVID-19 chest xray" and "Chest X-Ray Images (Pneumonia)" obtained from Kaggle. They assembled a dataset consisting of 160 COVID-19 patient images and 160 normal images, aiming to facilitate COVID-19 detection from chest X-ray images. These images were stored in the Databricks File System (DBFS) within the Databricks Workspace. Their experimentation took place in the Databricks Workspace utilizing a cluster configured with Databricks Runtime 6.4. They managed and accessed the chest X-ray images in DBFS, organizing them into distinct paths for COVID-19 and normal images. Two convolutional neural network-based models, InceptionV3 and ResNet50, underwent training and testing on the chest X-ray images using Apache Spark. They adopted a deep transfer learning approach, merging Deep Learning Pipelines and logistic regression. The dataset underwent partitioning into training and testing subsets to facilitate general principles learning while ensuring an accurate assessment of model performance. Deep Learning Pipelines enabled swift transfer learning on the Apache Spark cluster, employing a DeepImageFeaturizer to extract features from pre-trained CNN models for logistic regression. They evaluated the model's performance using metrics such as accuracy, F1-Score, weighted precision, and weighted recall. Both InceptionV3 and ResNet50 models exhibited high accuracy and performance across all tested indices. Their research underscores the efficacy of utilizing deep transfer learning methods within the Apache Spark framework for COVID-19 detection from chest X-ray images. The amalgamation of these techniques yielded advanced outcomes, highlighting the model's ability to accurately distinguish individuals with COVID-19 from those without it in X-ray images.

Çağın Polat et al. (2021) gathered three separate datasets from diverse origins, namely "ChestX-ray14," "COVID-19 image data collection," and "Chest X-ray collection from Indiana University," with the aim of aiding COVID-19 diagnosis through chest X-ray analysis. The ChestX-ray14 dataset consisted of 112,120 frontal-view chest radiographs from 30,805 patients, focusing particularly on those labeled with pneumonia. These labels were automatically generated using Natural Language Processing (NLP) techniques, boasting an accuracy rate of over 90%. The second dataset, COVID-19 image data collection, included 208 radiographs for COVID-19 cases and 41 for non-COVID-19 cases, albeit featuring some lossy images and adjustments by authors, potentially introducing bias. To counteract this potential bias, they proposed preprocessing methods. The third dataset, acquired from Indiana University, acted as a testing dataset for non-COVID-19 cases, featuring 50 pneumonia cases manually identified by experts. The collected data were categorized into non-COVID-19 pneumonia and COVID-19 pneumonia cases, displaying a noticeable imbalance in case numbers between these categories. To rectify this imbalance, they implemented strategies during transfer learning and optimization phases. Preprocessing entailed converting radiographs to JPEG format with varying compression levels and applying random adjustments to brightness, contrast, and sharpness to mitigate potential bias. Additionally, radiographs were colored using the colormap JET to ensure compatibility with pre-trained architectures designed for RGB inputs. Augmentation techniques, such as flipping, rotating, zooming, and adjusting brightness values, were employed to enrich dataset diversity. Due to the limited dataset size, they employed transfer learning, leveraging pre-trained architectures like ResNet, DenseNet, and VGG trained on ImageNet. Initially, convolutional layers were kept frozen, with only fully connected layers re-trained to distinguish COVID-19 pneumonia from non-COVID-19 pneumonia. Subsequently, fine-tuning was conducted on selected architectures to further optimize performance. The training process comprised three stages: exploring different architectures, fine-tuning, and evaluating activation maps. Performance evaluation criteria included accuracy, recall, and precision, with confusion matrices utilized to summarize model performance. In the initial transfer learning stage, various architectures exhibited similar performance levels, leading them to fine-tune specific architectures based on computational efficiency and performance in analogous medical studies.

CHAPTER 2. METHODOLOGY

2.1. Purpose and Objectives

The unprecedented emergence of COVID-19 has emphasized the urgent need for innovative strategies to mitigate its spread and impact. This thesis aims to leverage transfer learning techniques, particularly in medical image analysis, to create a predictive model capable of distinguishing between COVID-19 patients and healthy individuals. Utilizing Python and a dataset obtained from Kaggle containing diverse medical images, this study endeavors to enrich the repertoire of tools available for combatting the pandemic (Roy et al., 2022).

The primary aim is to develop and assess a robust machine learning model proficient in identifying COVID-19 infected individuals based on medical imaging data. Through transfer learning, which involves adapting pre-existing models to novel tasks, the focus is on leveraging insights from established models trained on extensive datasets to enhance the predictive model's performance. Key tasks include preprocessing medical images, fine-tuning pre-trained convolutional neural network (CNN) architectures, and evaluating the model's efficacy using appropriate metrics. Moreover, this research seeks to address specific sub-objectives within the broader framework of COVID-19 prediction using transfer learning methodologies. It aims to explore diverse CNN architectures and transfer learning strategies to identify the most suitable approach for the task at hand. Additionally, the thesis endeavors to examine how dataset characteristics, such as image resolution and class distribution, impact the model's performance and its ability to generalize. Efforts will also be directed towards optimizing hyperparameters and managing potential challenges like class imbalances to ensure the predictive model's robustness and reliability. Beyond the creation of the predictive model, this thesis aims to contribute to a deeper understanding of transfer learning applications in the medical field, particularly in infectious disease detection contexts. By elucidating the methodologies, challenges, and potential pitfalls associated with employing transfer learning for COVID-19 prediction, this research seeks to facilitate knowledge dissemination and inspire further exploration in this evolving area of study.

2.2. Overview of Transfer Learning

Transfer learning stands as a cornerstone technique in both machine learning and deep learning, wherein insights gleaned from addressing one problem are applied to tackle another related challenge. Its utility shines particularly bright in scenarios marked by scarce data or domain-specific intricacies. At the heart of transfer learning lie pre-trained models—neural network architectures honed on expansive datasets like ImageNet. Models such as EfficientNet, VGG, and MobileNet encapsulate fundamental features from diverse image categories, serving as foundational frameworks for subsequent customization to specific tasks. Fine-tuning emerges as a prevalent strategy within transfer learning, entailing the adjustment of pre-trained model parameters through additional training on a target dataset. This process enables the model to adapt its acquired representations to the subtleties of the target task, thereby amplifying performance and enhancing its ability to generalize. Another avenue, feature extraction, involves capturing intermediate representations learned by pre-trained models and employing them as input features for a task-specific classifier. This methodology circumvents the need for extensive retraining while still leveraging the benefits of the acquired representations (Rahmani et al., 2022). The repertoire of transfer learning methodologies extends to domain adaptation, which focuses on tailoring models trained on one domain to effectively operate in another domain with distinct distributions. This proves invaluable when labeled data in the target domain is scarce or inaccessible. Meanwhile, multi-task learning facilitates concurrent model training on multiple interconnected tasks, enabling the model to leverage shared knowledge and acquire task-specific insights more efficiently. This fosters resilience and adaptability across a spectrum of tasks. In the realm of predicting COVID-19 through medical image processing, transfer learning assumes paramount significance in elevating model performance and scalability (Figure 2.1). By harnessing pre-trained models trained on expansive image datasets, researchers can extract meaningful features from medical images and discern patterns indicative of COVID-19 infection or health status. Moreover, transfer learning streamlines the development of robust and interpretable models capable of accommodating variations in imaging protocols, equipment, and patient demographics. This paves the way for the deployment of dependable diagnostic tools for COVID-19 screening and patient management, thereby contributing to the global endeavor to combat the pandemic. As the field continues its evolution, transfer learning is poised to occupy an increasingly pivotal role in shaping the trajectory of medical imaging and disease diagnosis (Mwaniki, 2023; Laddha et al., 2022). Figure 2.1 illustrates the impact of transfer learning in COVID-19 diagnosis.

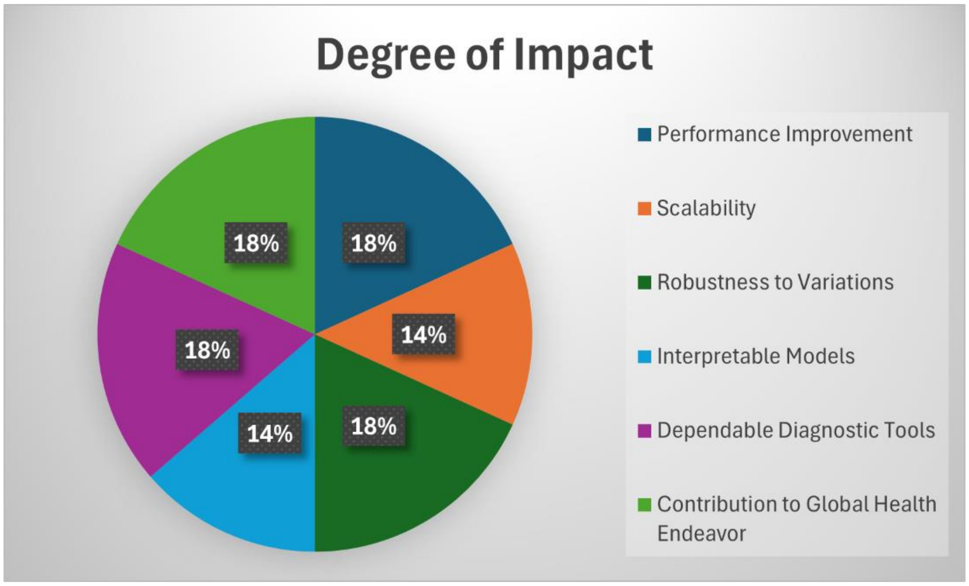

Figure 2.1. The impact of transfer learning in COVID-19 diagnosis

2.3. Understanding Medical Imaging Data

Medical imaging serves as a cornerstone in modern healthcare, providing clinicians with invaluable insights into internal structures and anomalies within the human body. Digital imaging technologies like X-ray, computed tomography (CT), magnetic resonance imaging (MRI), and positron emission tomography (PET) generate vast amounts of medical image data daily. Understanding the intricacies of this data is essential for developing accurate predictive models for COVID-19 using transfer learning techniques (Rudroff, 2024).

These modalities offer distinct perspectives on anatomical and physiological structures. X-ray imaging yields two-dimensional projections suitable for detecting lung abnormalities like pneumonia or COVID-19-associated lesions (Wibisono et al., 2019). CT scans provide cross-sectional images with higher resolution, allowing for detailed lung morphology visualization (Lu et al., 2016). MRI offers detailed soft tissue images, valuable for assessing brain and musculoskeletal disorders (Soppari et al., 2024). PET imaging visualizes metabolic activity within tissues, aiding in disease diagnosis and staging (Hess et al., 2014). The sheer volume of data can overwhelm traditional computational methods. Variability in resolution, orientation, and contrast requires preprocessing for standardization. Interpreting medical images demands domain-specific expertise to detect subtle abnormalities. Ensuring patient privacy and data security is crucial due to sensitive information contained in medical images (Shachar, 2024).

Transfer learning offers a solution by leveraging pre-trained deep learning models for medical imaging tasks. This technique adapts models trained on large-scale datasets to overcome limitations of small and heterogeneous medical image datasets. In COVID-19 prediction, transfer learning enables the adaptation of pre-trained models to classify chest X-ray or CT images, accelerating model development and enhancing performance. TensorFlow, coupled with the Keras API, provides a flexible framework for implementing transfer learning in medical imaging. Keras simplifies model building and training with its high-level interface, ideal for rapid prototyping. TensorFlow's integration with Keras Applications grants access to pre-trained models like EfficientNet, VGG, and MobileNet. Fine-tuning these models on medical imaging datasets captures disease-specific features relevant to COVID-19 prediction. Extensive documentation and community support ensure developers have resources and guidance throughout the model development process (Chola et al., 2022).

2.4. Challenges in COVID-19 Detection

The outbreak of COVID-19 has posed unprecedented challenges to healthcare systems worldwide, highlighting the urgent need for swift and accurate diagnostic methods to curb its spread and impact. Despite advancements in medical imaging and machine learning, detecting COVID-19 remains a complex endeavor beset by numerous obstacles (see Table 2.1). We delve into these challenges and examine how transfer learning techniques, particularly employing TensorFlow with the Keras API, can offer solutions. One of the foremost hurdles in COVID-19 detection via medical imaging is the variability in image characteristics observed across different patients and imaging modalities. Images sourced from diverse origins may vary significantly in resolution, contrast, and noise levels, complicating the task for conventional machine learning algorithms. Transfer learning, by harnessing pre-trained models and adapting them to new tasks, can mitigate this challenge. By assimilating meaningful representations from a wide array of image data, transfer learning enhances the resilience of detection models (Jafari et al., 2022).

Table 2.1. Comparison of model performance

Model Type	Accuracy	Sensitivity	Specificity	Remarks
Conventional ML	High	Moderate	Moderate	Limited by variability in image features
Transfer Learning	Very High	High	High	Enhanced resilience to image variability

Another formidable challenge lies in the scarcity of labeled data for training deep learning models tailored to COVID-19 detection. Amassing large-scale annotated datasets for such imaging studies is arduous due to privacy constraints, access limitations, and the laborious nature of manual labeling. Here, transfer learning emerges as a beacon of hope. It facilitates the transfer of knowledge from pre-trained models trained on expansive datasets to the realm of COVID-19 detection. Through fine-tuning on a smaller dataset of COVID-19 images, transfer learning enables the creation of precise and dependable detection models, even in the face of limited labeled data (Yang, 2024).

Class imbalance and data skew pose additional hurdles in medical imaging datasets. The overrepresentation of one class, such as healthy patients, compared to another, like COVID-19 positive patients, can bias models and lead to subpar performance, exacerbated by differences in data distribution between training and testing sets. Transfer learning methods, coupled with techniques like data augmentation and class-weighted loss functions, offer remedies to these issues. By addressing class imbalance and data skew, transfer learning enhances the generalization performance of detection models across various classes (Figure 2.2.).

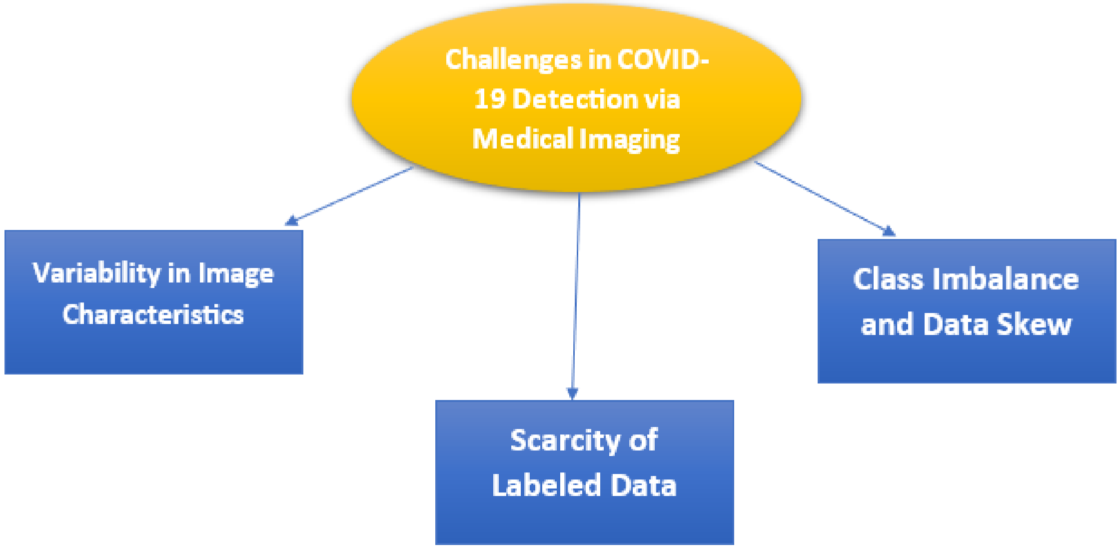

Figure 2.2. *Challenges in COVID-19 Detection via Medical Imaging*

Ensuring the interpretability and explainability of COVID-19 detection models is paramount, particularly in medical imaging applications where AI-driven decisions directly impact patient care (Röösli et al., 2022). Deep learning models, especially those trained using transfer learning, often exhibit intricate and opaque decision-making processes, complicating interpretation. Guaranteeing the interpretability and explainability of these models is essential to garner the trust of healthcare practitioners and seamlessly integrate them into clinical workflows. Techniques such as feature visualization, saliency mapping, and model explanation methods serve to enhance the interpretability of transfer learning-based detection models, empowering clinicians to comprehend and rely on their predictions (Nhlapho et al., 2024).

2.5. Publicly Available Datasets for COVID-19 Chest X-ray Images

The availability of publicly accessible datasets plays a pivotal role in advancing research endeavors, particularly in the domain of medical image analysis. Amidst the ongoing COVID-19 pandemic, the demand for high-quality chest X-ray datasets for developing predictive models has surged. One of the notable datasets utilized in numerous studies is the dataset curated by Mrinal Tyagi et al. (2022). This dataset comprises chest X-ray images sourced from diverse medical institutions, encompassing a spectrum of COVID-19 cases along with other pathologies such as Normal, Lung Opacity (LO), and Viral Pneumonia (VP). The authors have meticulously balanced the dataset by employing both under-sampling and over-sampling techniques, ensuring representative distribution across classes. Furthermore, the authors have applied novel image processing techniques, including a Singular Value Decomposition (SVD)-based approach, to enhance the quality and diversity of images within the dataset.

2.5.1. Ethical Considerations

The use of medical data, including chest X-ray images, raises important ethical considerations related to patient privacy, consent, and data security. Adhering to ethical guidelines and regulations is paramount to ensuring the responsible use of data and protecting the rights and confidentiality of individuals involved.

2.5.1.1. Patient Privacy

Protecting patient privacy is a fundamental ethical principle in medical research and data science. Chest X-ray images may contain sensitive information about individuals' health status, demographics, and medical history. Therefore, it's essential to de-identify or anonymize the data to prevent the identification of individuals and minimize privacy risks.

2.5.1.2. Informed Consent

Obtaining informed consent from patients or participants is essential when using medical data for research purposes. Researchers must ensure that individuals understand the nature of the research, its potential risks and benefits, and how their data will be used and protected. Informed consent helps uphold autonomy and respect for individuals' rights and ensures transparency in the research process.

2.5.1.3. Institutional Review Board (IRB) Approval

Many research institutions require approval from an Institutional Review Board (IRB) or Ethics Committee before conducting research involving human subjects or medical data. The IRB evaluates research proposals to ensure compliance with ethical standards, patient safety, and regulatory requirements. Researchers must obtain IRB approval before collecting, analyzing, or sharing medical data to ensure ethical conduct and regulatory compliance (Wicks & Chiauzzi, 2019).

Authors of the dataset provided a citation requirement for the use of their dataset, indicating their intention to receive credit for their work.

2.6. Preprocessing of Chest X-ray Images

The preprocessing of chest X-ray images plays a crucial role in enhancing the quality and balance of the dataset, ensuring optimal performance of machine learning models for classification tasks. In this section, we delve into the preprocessing techniques employed by the authors to address the imbalanced nature of the dataset and enhance its features.

2.6.1. Dataset Overview

The dataset utilized in this study comprises chest X-ray images collected from individuals diagnosed with Coronavirus disease (COVID-19), caused by the SARS-CoV-2 virus. This infectious disease manifests with varying degrees of severity, with some individuals experiencing mild symptoms while others may develop serious respiratory complications. The dataset initially consisted of four classes: Covid, Normal, Lung Opacity (LO), and Viral Pneumonia (VP), with varying numbers of images per class.

2.6.2. Balancing the Dataset

The original dataset exhibited significant class imbalance, with a disproportionate distribution of images across the different classes. To mitigate this issue, the authors employed a combination of under-sampling (Random Under-Sampling, RUS) and over-sampling (data augmentation) techniques through image processing methods.

2.6.3. SVD-Based Image Processing

A novel Singular Value Decomposition (SVD)-based image processing technique was employed specifically for the minor classes, VP and Covid. This technique aimed to generate synthetic images with slightly altered luminance and contrast characteristics, effectively augmenting the dataset and addressing the imbalance.

2.6.4. Contrast Limited Adaptive Histogram Equalization (CLAHE)

To enhance the features of the entire dataset, Contrast Limited Adaptive Histogram Equalization (CLAHE) with a parameter of 0.5 was applied uniformly across the images. Additionally, a value of 1.0 was selectively chosen for the VP class to prevent excessive contrast enhancement. This step ensured that the features extracted from the chest X-ray images were optimized for subsequent analysis.

2.6.5. Augmented Dataset Analysis

In the augmented chest X-ray (CXR) dataset, the number of images per class varied slightly, reflecting the unique characteristics and distribution of images within each class. Notably, the Covid class exhibited significantly higher intra-class variance compared to other classes. Consequently, a nuanced approach was adopted to determine the number of augmented images for this class, aiming for improved convergence of convolutional neural network (CNN) models.

2.6.6. Intra-Class Variance Analysis

The authors conducted a correlation coefficient analysis to quantify the mean intra-class variance for each class. This analysis informed the selection of the number of augmented images, ensuring a balance between class representation and intra-class variance. As a result, the augmented dataset achieved a more equitable distribution, facilitating robust model training and evaluation (Roy et al., 2022).

2.7. Data Augmentation Techniques

Data augmentation plays a pivotal role in enhancing the performance and robustness of machine learning models, particularly in scenarios where the available dataset is limited or imbalanced. In the context of this thesis, where I aimed to develop a model using transfer learning techniques with medical images for identifying ill and healthy patients, employing effective data augmentation techniques becomes imperative. Data augmentation involves artificially generating new training samples by applying a variety of transformations to existing data. This process not only increases the size of the dataset but also introduces diversity, thereby enabling the model to generalize better to unseen data. In medical imaging tasks like mine, where obtaining labeled data is often challenging and expensive, data augmentation becomes particularly crucial for training accurate and robust models (Nair et al., 2024). Several techniques are commonly used in data augmentation, including flip, scaling, shear, and zoom. Each technique introduces specific variations to the input data, thereby enriching the dataset with diverse examples for training. There are advantages of using them. Augmented data exposes models to a wider range of variations, enabling them to generalize better to unseen data and diverse real-world scenarios. Augmentation expands the dataset, providing more training examples for the model to learn from, which often leads to better performance, especially in scenarios with limited original data. By regularizing the model and preventing it from memorizing the training data, data augmentation helps mitigate overfitting, leading to models that generalize well to new data (Dai et al., 2022).

Each data augmentation technique served a specific purpose in enhancing the model's learning process. For instance, rescaling normalized pixel values to ensure consistent feature scales across the dataset, while shear range and zoom range introduced variations in orientation and scale, respectively, augmenting the dataset with diverse examples. Horizontal flip added further variability by flipping images horizontally, aiding the model in learning invariant features.

2.7.1. Rescaling

This technique involves resizing the input images uniformly by a certain factor. It's commonly used to normalize images to a standard size before feeding them into a neural network, which can improve training efficiency and convergence by reducing computational overhead and ensuring consistency in feature extraction across different resolutions. I applied rescaling to standardize the pixel values of the images between 0 and 1. This normalization step facilitated faster convergence during training and ensured numerical stability.

2.7.2. Shear Range

Shearing involves shifting one part of the image along a certain axis, creating a "stretching" effect. Shear range refers to the maximum angle or magnitude by which the image can be sheared. By applying shear transformations within a specified range, data augmentation can introduce variability in object orientations, helping the model generalize better to variations in real-world scenarios, such as tilted or skewed objects.

2.7.3. Zoom Range

Zoom range determines the extent to which the input image can be magnified or shrunk. During data augmentation, randomly zooming in or out on images helps simulate the effect of varying distances between the camera and objects in the scene. This can enhance the model's ability to recognize objects at different scales and improve its robustness to changes in perspective and viewing angles.

2.7.4. Horizontal Flip

This augmentation technique involves flipping images horizontally along the vertical axis. It's particularly useful for tasks where object orientation or left-right symmetries are not relevant, such as object classification. By introducing mirror images of the training data, the model learns to be invariant to horizontal flips, thereby enhancing its ability to generalize to unseen data and improving overall performance (See Table 2.2). When implementing data augmentation techniques, it is essential to strike a balance between introducing sufficient diversity into the dataset and preserving the integrity of the original images. Additionally, it is crucial to ensure that the augmented images remain clinically relevant and do not introduce artifacts or distortions that could compromise the model's performance. Furthermore, the choice of data augmentation techniques should be guided by the specific characteristics of the medical imaging task at hand and the potential variations present in the dataset. Experimentation with different augmentation strategies and parameters is often necessary to determine the most effective augmentation pipeline for training the model (Dai et al., 2022).

Table 2.2. Comparison of data augmentation techniques in medical image classification

Data Augmentation Technique	Purpose/Effect	Implementation Details
Rescaling	Normalize images to a standard size	Resize input images uniformly by a certain factor
Shear Range	Introduce variability in object orientations	Shift parts of the image along a certain axis
Zoom Range	Simulate varying distances between camera and objects	Magnify or shrink input images randomly

Data Augmentation Technique	Purpose/Effect	Implementation Details
Horizontal Flip	Introduce mirror images for left-right invariance	Flip images horizontally along the vertical axis

2.8. Transfer Learning Setup

Transfer learning has emerged as a powerful technique in the realm of machine learning, allowing models to leverage knowledge gained from one task to improve performance on another related task. At its core, transfer learning involves the reusability of knowledge or representations learned from a source domain to a target domain. This approach is particularly valuable when the target task has limited or insufficient labeled data for training a model from scratch. Transfer learning operates on the principle of transferring knowledge from a pre-trained model to a new task, thereby accelerating the learning process and potentially improving the performance of the target model (Fumagalli et al., 2020). In traditional machine learning approaches, models are trained from scratch on a specific dataset, requiring a substantial amount of labeled data and computational resources. However, in transfer learning, instead of starting from random initialization, the model initializes with parameters learned from a related task or dataset. These pre-trained models have already learned meaningful representations from vast amounts of data and can effectively capture high-level features that are transferable across tasks.

The significance of transfer learning lies in its ability to address the challenges of data scarcity and domain shift commonly encountered in real-world applications. By leveraging pre-existing knowledge encoded in pre-trained models, transfer learning enables models to generalize better to new tasks and adapt more quickly to changes in the input distribution. This not only reduces the need for large labeled datasets but also enhances the robustness and scalability of machine learning systems (Wang & Chen, 2023).

2.8.1. VGG16

VGG16 stands as a testament to the expertise of the Visual Geometry Group at the University of Oxford, having played a pivotal role in securing victory during the intense competition of ImageNet in 2014. Widely hailed as a pinnacle achievement in vision model architectures, VGG16 boasts a meticulous design featuring thirteen convolutional layers, each intricately woven with 3 × 3 filters and a stride of 1. It employs the technique of same padding to maintain spatial dimensions and integrates max pool layers with a 2 × 2 filter and a stride of 2, ensuring effective feature extraction. This meticulous sequence of convolution and max pool layers remains consistent throughout the architecture, showcasing a thoughtful approach to information processing. Furthermore, VGG16 is fortified with two fully-connected layers and a discerning output layer, culminating in a robust network architecture poised for diverse tasks. The journey of VGG16 through training was no small feat, traversing a vast dataset comprising 1.2 million images meticulously categorized into 1,000 classes. Each iteration of training served to refine the network's understanding, honing its ability to discern patterns and features essential for accurate classification (Akinyelu & Blignaut, 2022).

2.8.2. InceptionV3

InceptionV3, a prominent member of the inception family within the realm of convolutional neural network (CNN) architectures, stands out for its extensive depth, boasting a total of 48 layers. This architecture distinguishes itself by implementing various techniques to enhance performance and efficiency. For instance, it integrates label smoothing and an auxiliary classifier as forms of regularization, ensuring stable training and robustness against overfitting. Moreover, the utilization of factorized 7x7 convolutions serves a dual purpose: it reduces the number of parameters required for the model while maintaining computational efficiency. One of the key innovations of InceptionV3 lies in its architectural design, where batch normalization plays a crucial role. Positioned strategically between the auxiliary classifier and the fully-connected layer, batch normalization acts as an additional regularizer, contributing to the overall stability and convergence of the network. InceptionV3's prowess extends beyond its architectural intricacies; it has been pre-trained on the vast ImageNet dataset, a benchmark in image classification tasks. Through this pre-training process, the network has acquired a rich understanding of visual features and patterns inherent in diverse images. Central to InceptionV3's architecture are its convolutions, which blend various kernel sizes (1x1, 3x3, and 5x5) with max pooling layers. This amalgamation results in a network capable of effectively capturing intricate details and hierarchical structures within images, facilitating robust and accurate classification (Akinyelu & Blignaut, 2022; Himel & Islam, 2024).

2.8.3. EfficientNet

EfficientNet stands out as a highly efficient architecture for image classification, employing a unique approach known as compound scaling (Reza et al., 2021). Rather than independently adjusting depth, width, or resolution, EfficientNet strategically scales all three dimensions together, resulting in improved performance. This technique optimizes the network's depth, width, and resolution simultaneously, achieving superior accuracy and efficiency (Reza et al., 2021).

Introduced in 2019, EfficientNet offers eight different versions varying in parameters and complexity. Modifications to its core layers enhance precision. Recent research has explored all eight models, often focusing on data preprocessing or adjustments to end layers for classification. The model's parameter reduction goal has prompted comparisons with other architectures, particularly in the context of COVID-19 detection systems. EfficientNet emerges as a promising candidate for comparison, potentially offering superior performance with fewer parameters compared to older, widely-used models (Mozaffari et al., 2023).

2.8.4. MobileNet

MobileNet stands out as an intricate neural network design meticulously crafted to cater to the constraints of mobile and embedded devices, ensuring efficient processing. Its innovation lies in the implementation of depth-wise separable convolution, a technique adept at diminishing computational demands without compromising accuracy. This unique feature renders MobileNet ideal for real-time applications operating within the confines of limited computing resources (Himel & Islam, 2024).

2.9. Architecture Design of CNN for COVID-19 Prediction

Convolutional Neural Networks (CNNs) represent a class of deep learning models that excel in processing grid-like data, particularly images. Unlike traditional neural networks, CNNs leverage convolutional and pooling layers to automatically learn and extract features from input images, making them highly suitable for image classification tasks. The core idea behind CNNs is inspired by the human visual system, where neurons in the visual cortex respond to specific stimuli within receptive fields. Similarly, in CNNs, each neuron in a convolutional layer is responsible for detecting particular features within its receptive field, which are then hierarchically combined to form higher-level representations. The suitability of CNNs for image classification stems from their ability to capture spatial hierarchies of features. As information passes through successive layers of convolution and pooling, the network learns to identify low-level features such as edges and textures in the early layers, gradually progressing to more abstract and complex features in deeper layers.

The components and layers of the model include convolutional layers, pooling layers, a flatten layer, and fully connected layers:

Convolutional Layers: These layers are the building blocks of CNNs and play a fundamental role in feature extraction. A convolutional layer applies a set of learnable filters (also known as kernels) to the input image, performing a convolution operation. Each filter detects specific patterns or features within the input image by sliding over it and computing element-wise multiplications followed by summation. In the architecture designed for COVID-19 prediction, I employed three convolutional layers. The first layer consists of 32 filters, followed by 64 and 128 filters in the subsequent layers. Increasing the number of filters allows the network to capture increasingly complex features as the information progresses through the layers.

Pooling Layers: Pooling layers are used to reduce the spatial dimensions of the feature maps produced by the convolutional layers while retaining important information. The most common pooling operation is max-pooling, where the maximum value within each region of the feature map is selected. This downsampling operation helps in reducing computational complexity and controlling overfitting by providing a form of spatial hierarchy abstraction. In my CNN architecture, max-pooling layers with a pool size of (2, 2) are inserted after each convolutional layer. This reduces the spatial dimensions of the feature maps by a factor of two along both the width and height dimensions, effectively reducing the number of parameters in subsequent layers.

Flatten Layer: After the convolutional and pooling layers, the Flatten layer is used to convert the two-dimensional feature maps into a one-dimensional vector. This transformation is necessary to feed the output of the convolutional and pooling layers into the fully connected layers, which require one-dimensional input.

Fully Connected Layers: These layers are typical neural network layers where each neuron is connected to every neuron in the preceding and succeeding layers. Fully connected layers are responsible for learning global patterns and relationships in the extracted features. In my architecture, I included a single dense layer with 128 neurons, followed by a dropout layer and a final dense layer with a single neuron for binary classification (COVID vs. non-COVID). The use of ReLU activation functions in the dense layers introduces non-linearity, allowing the model to learn complex relationships between features. Additionally, dropout regularization is applied to the first dense layer with a dropout rate of 0.5 to prevent overfitting by randomly dropping out a fraction of the neurons during training.

The design of CNN architectures for medical image analysis, particularly for COVID-19 detection, requires special considerations due to the unique characteristics of medical imaging data. Below are the specific adaptations made to address these challenges. Medical images, such as X-rays and CT scans, often come in various sizes and resolutions. However, for efficient processing and compatibility with pre-trained models, it is essential to standardize the input image dimensions. In my architecture, I set the input dimensions to 224x224 pixels, which is a common size used in many pre-trained CNN models trained on large-scale datasets like ImageNet. By resizing the images to a consistent size, I ensured that the CNN model can effectively learn and extract features from the input images without being influenced by variations in image dimensions. The choice of activation function in CNN architectures significantly impacts the model's ability to learn and generalize from the data. Rectified Linear Unit (ReLU) activation functions are commonly used in convolutional layers due to their simplicity and effectiveness in introducing non-linearity. ReLU activation functions replace negative pixel values with zero, effectively introducing a threshold below which the neuron is inactive. This helps in mitigating the vanishing gradient problem and accelerating the convergence of the training process. Overfitting is a common challenge in deep learning models, particularly when dealing with limited training data. Dropout regularization is a technique used to prevent overfitting by randomly deactivating a fraction of neurons during training. In my architecture, a dropout layer with a dropout rate of 0.5 is applied after the first fully connected layer. During training, each neuron in the dropout layer has a probability of 0.5 of being temporarily removed, forcing the network to learn more robust and generalizable features.

Transfer learning is a powerful technique in deep learning, where knowledge gained from training on one task is leveraged to improve performance on a different but related task. In the context of CNNs, transfer learning involves initializing the model with pre-trained weights obtained from a large-scale dataset and fine-tuning the model on a new dataset specific to the target task. Transfer learning offers significant advantages in the realm of COVID-19 prediction using convolutional neural networks (CNNs). Firstly, it facilitates faster convergence during training by leveraging pre-trained models' learned representations from extensive datasets like ImageNet. These pre-trained weights provide the model with a starting point for learning relevant features for COVID-19 prediction, reducing training time and computational resources. Additionally, transfer learning enhances the model's generalization capabilities by allowing it to leverage knowledge acquired from diverse images across different domains. Fine-tuning the pre-trained weights on the COVID-19 dataset is crucial for adapting the model's learned representations to capture specific features and patterns indicative of COVID-19 infection in medical images. This process improves the model's performance and robustness in accurately predicting COVID-19 cases from X-ray or CT scan images. The implementation of transfer learning can vary depending on factors such as the availability of labeled data and computational resources. In my architecture, I adopted a common transfer learning approach by utilizing pre-trained weights from established CNN architectures like VGG, MobileNet, or Inception. Subsequently, I fine-tuned the model on the COVID-19 dataset to optimize its performance for the specific prediction task at hand.

CHAPTER 3. IMPLEMENTATION AND TRAINING

Fine-tuning pre-trained models represents a pivotal strategy in the domain of transfer learning, particularly when applied to tasks such as COVID-19 prediction using medical images. In this section, we delve into the definition, process, strategies, optimization techniques, considerations, and practical implementation aspects of fine-tuning pre-trained convolutional neural networks (CNNs) for COVID-19 prediction.

Fine-tuning, within the context of transfer learning, embodies the process of adapting pre-trained models to new tasks by adjusting their parameters. This process capitalizes on the knowledge encoded in the pre-trained weights, obtained from training on a large dataset for a related task, and tailors the model's parameters to suit the intricacies of the target task. Fine-tuning is essential for enabling the model to learn task-specific features while leveraging the wealth of information captured during pre-training (Ramdan et al., 2020).

Adapting pre-trained CNNs for COVID-19 prediction encompasses several crucial steps:

1. **Network Architecture Modification**: Fine-tuning often necessitates adjustments to the architecture of the pre-trained CNN to align with the specific requirements of the prediction task. This may entail modifying the number of layers, adding or removing convolutional or pooling layers, or altering the size of the fully connected layers (Kambale et al., 2024). The goal is to tailor the architecture to effectively extract relevant features from medical images for accurate prediction of COVID-19 status.

2. **Selection of Trainable Layers**: Not all layers of the pre-trained CNN need to be fine-tuned. Depending on the similarity between the pre-training task and the target task, only a subset of layers may be updated during fine-tuning. Typically, lower layers responsible for learning generic features, such as edges and textures, are frozen, while higher layers capturing more task-specific information, such as disease-related patterns, are fine-tuned to adapt to the COVID-19 prediction task.

3. **Adjustment of Hyperparameters**: Fine-tuning also involves tuning hyperparameters such as learning rate, batch size, and dropout rate to optimize model performance on the target

task. Hyperparameter tuning is critical for achieving convergence during training and preventing issues such as underfitting or overfitting (Kambale et al., 2024).

Several transfer learning strategies are commonly employed during the fine-tuning process:

1. **Freezing Early Layers**: Freezing the early layers of the pre-trained CNN helps preserve generic features learned from the pre-training task. By keeping these layers fixed, the model can focus on learning task-specific features from the COVID-19 prediction dataset without overwriting the valuable representations captured in the lower layers. This prevents catastrophic forgetting and enables efficient transfer of knowledge from the pre-trained model.

2. **Fine-tuning Deeper Layers**: Deeper layers of the pre-trained CNN, closer to the output layer, tend to capture more task-specific information relevant to COVID-19 prediction. Fine-tuning these layers allows the model to adapt its representations to the nuances of the target task, thereby enhancing its predictive performance. Fine-tuning deeper layers enables the model to learn intricate patterns and correlations present in the COVID-19 dataset, leading to improved generalization and discriminative capability (Philippi et al., 2023).

During fine-tuning, optimization algorithms are employed to update the parameters of the pre-trained model and minimize the loss function. Common optimization techniques include:

1. **Stochastic Gradient Descent (SGD)**: SGD is a fundamental optimization algorithm used to iteratively update the model parameters based on the gradients computed from a random subset of training samples. While SGD is simple and computationally efficient, it may suffer from slow convergence or oscillations in the parameter space. Variants of SGD, such as mini-batch SGD and momentum SGD, address some of these limitations and improve training stability.

2. **Adaptive Learning Rates**: Adaptive learning rate algorithms, such as Adam and RMSprop, dynamically adjust the learning rate for each parameter based on past gradients. These algorithms adaptively scale the learning rates based on the magnitude and direction of the gradients, leading to faster convergence and improved generalization performance. Adam,

in particular, combines the advantages of adaptive learning rates and momentum to achieve superior optimization performance across a wide range of tasks (Ioannou et al., 2023).

Fine-tuning pre-trained models for COVID-19 prediction poses several considerations and challenges:

1. **Overfitting**: Fine-tuning on a small dataset may lead to overfitting, where the model memorizes noise in the training data rather than learning generalizable features. Regularization techniques such as dropout, weight decay, and early stopping are commonly employed to mitigate overfitting and improve the model's ability to generalize to unseen data. Additionally, techniques like data augmentation can help increase the diversity of the training data and prevent overfitting by exposing the model to a broader range of variations.

2**. Domain Shift**: Pre-trained models may have been trained on datasets with different characteristics than the target COVID-19 prediction dataset. Domain adaptation techniques, such as domain adversarial training or domain-specific normalization layers, may be necessary to align the feature distributions between the source and target domains and prevent performance degradation due to domain shift. By minimizing the distribution discrepancy between the pre-training and fine-tuning datasets, domain adaptation techniques enable the model to generalize better to the target task.

3. **Data Augmentation**: Data augmentation techniques play a crucial role in enhancing the diversity of the training data and improving the model's robustness to variations in input images. Common data augmentation techniques include rotation, scaling, translation, flipping, and elastic deformation. By introducing variations in the training data, data augmentation helps expose the model to a broader range of visual cues and enhances its ability to learn invariant representations. However, care must be taken to ensure that augmented images remain clinically relevant and do not introduce unrealistic distortions that could compromise the model's performance (Helander, 2021).

Implementing fine-tuning for COVID-19 prediction entails several practical considerations:

1. **Software Frameworks**: Popular deep learning frameworks such as TensorFlow and PyTorch provide comprehensive APIs for fine-tuning pre-trained models. These frameworks offer high-level abstractions for building and training models, as well as efficient implementations of optimization algorithms and evaluation metrics. Leveraging pre-existing implementations of popular CNN architectures and fine-tuning strategies can significantly streamline the development process and accelerate model deployment (Rao, 2023).

2. **Computational Resources**: Fine-tuning pre-trained models often requires significant computational resources, including powerful GPUs or TPUs for training and validation. Cloud computing platforms such as Google Cloud Platform (GCP) and Amazon Web Services (AWS) offer scalable infrastructure for running deep learning experiments and accessing specialized hardware accelerators. By provisioning resources on-demand and parallelizing computations across multiple devices, cloud-based environments enable researchers to train and evaluate complex models more efficiently (Arif, 2020).

3. **Data Management and Preprocessing**: Managing and preprocessing the COVID-19 prediction dataset is a critical aspect of fine-tuning pre-trained models. This may involve tasks such as data cleaning, normalization, and augmentation to ensure the dataset is well-structured and representative of the target task. Tools and libraries for data manipulation and preprocessing, such as pandas, NumPy, and OpenCV, can facilitate these tasks and streamline the data pipeline (Singh, 2022). Additionally, storing and versioning the dataset using platforms like Google Cloud Storage or AWS S3 ensures data integrity and reproducibility across experiments (Ferrua, 2023).

4. **Model Evaluation and Validation**: Evaluating the performance of fine-tuned models on the COVID-19 prediction task requires careful design and execution of validation experiments. This involves splitting the dataset into training, validation, and test sets, selecting appropriate evaluation metrics, and conducting cross-validation to assess model generalization (See Figure 3.1.). Techniques such as k-fold cross-validation and stratified sampling help mitigate the effects of data imbalance and ensure robust performance estimation. Furthermore, visualizing model predictions and diagnostic metrics using tools like Matplotlib or

TensorBoard aids in interpreting model behavior and identifying areas for improvement (Mekruksavanich & Jitpattanakul, 2022; Abu Lekham et al., 2022).

My model performs image data preprocessing using ImageDataGenerator to prepare the data for training and testing. This includes:

- **Rescaling**: Both the training and testing datasets are rescaled by a factor of 1/255 to normalize the pixel values.
- **Training Augmentation**: Transformations like shearing (with a shear range of 0.2), zooming (with a zoom range of 0.2), and horizontal flipping are applied to artificially increase the training data size and improve model robustness. Data augmentation is skipped during testing to ensure a more realistic evaluation of the model's ability to handle unseen data. Augmenting test data might introduce variations not present in real-world images, potentially biasing the results. This step helps focus on the model's ability to generalize to new examples.

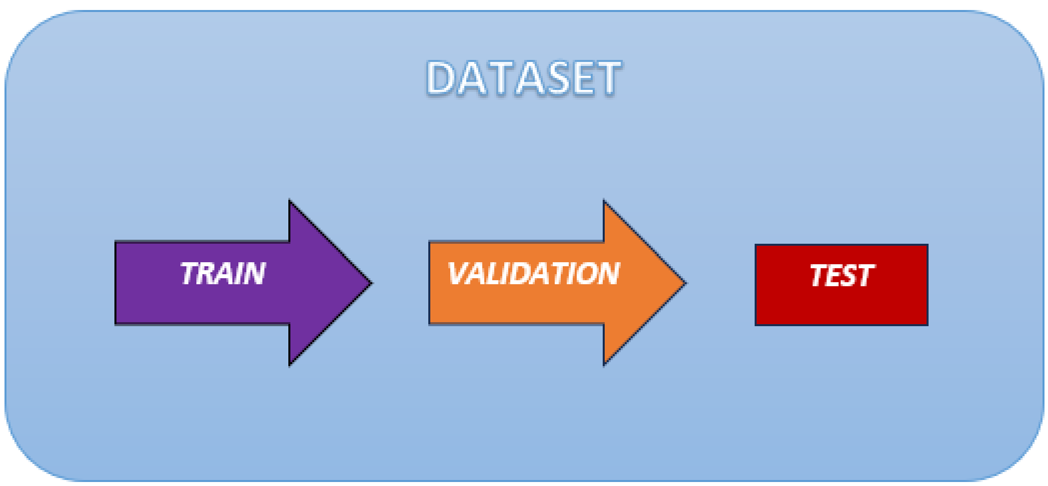

Figure 3.1. *Preparation of dataset*

3.1. Training Process

Transfer learning is a potent method within the realm of deep learning, wherein insights gleaned from addressing one problem are employed to tackle another problem that shares a similar context. Specifically concerning image classification endeavors, transfer learning entails utilizing pre-existing convolutional neural network (CNN) models that have been trained on extensive datasets like ImageNet. These models are then adjusted to suit a fresh task with a more limited dataset. Within this section, I detailed the application of transfer learning utilizing four well-known pre-trained models: VGG16, EfficientNet, InceptionV3, and MobileNet.

Before delving into the specifics of implementation, it's vital to carefully select the most suitable pre-trained models for the given task. The decision on which pre-trained models to utilize hinges on several factors such as the task's complexity, available computational resources, and the size of the dataset at hand. In this particular implementation, I've opted for four widely recognized pre-trained models:

1. VGG16: Renowned for its simplicity and efficacy, VGG16 comprises 16 convolutional layers and demonstrates remarkable performance across various image classification tasks.

2. EfficientNet: EfficientNet implements compound scaling, strategically adjusting depth, width, and resolution together, leading to improved accuracy and efficiency in image classification tasks.

3. InceptionV3: InceptionV3 adopts a more intricate architecture featuring inception modules, aiding in capturing spatial hierarchies of features across diverse scales.

4. MobileNet: Tailored for mobile and embedded vision applications, MobileNet offers a lightweight architecture well-suited for environments with constrained resources.

Figure 3.2 illustrates the main components and interactions of the CNN models developed for COVID-19 prediction using transfer learning. It outlines data preparation, model training, evaluation, and visualization stages.

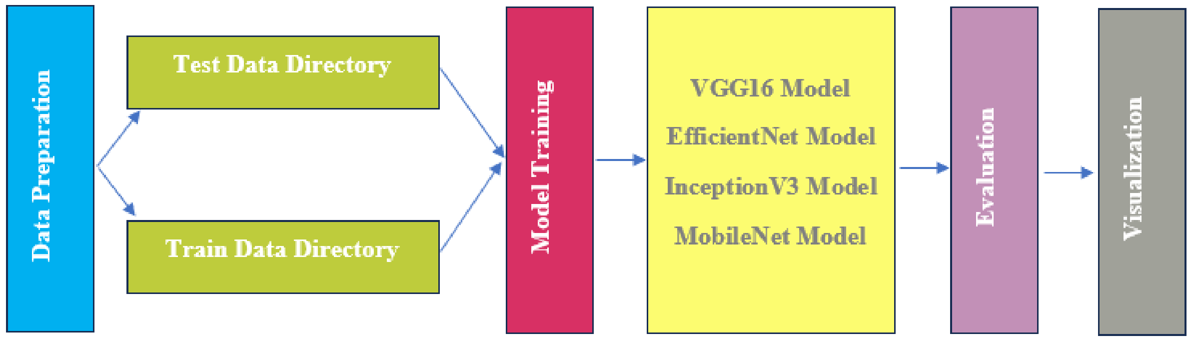

Figure 3.2. *Model Architecture Diagram*

To implement transfer learning, I leveraged the pre-trained models provided by the Keras library, which are already trained on the ImageNet dataset. Using Keras, I imported the pre-trained models while excluding the fully connected layers (top layers) to obtain the convolutional base of each model. This is achieved by setting the *include_top* parameter to *False* during model instantiation (see Figure 3.3.).

```
from tensorflow.keras.applications import VGG16, EfficientNet,
    InceptionV3, MobileNet

# Load pre-trained models
vgg16_base_model = VGG16(weights='imagenet', include_top=False,
    input_shape=(img_width, img_height, 3))
efficientnet_base_model = EfficientNet(weights='imagenet', include_top
    =False, input_shape=(img_width, img_height, 3))
inceptionv3_base_model = InceptionV3(weights='imagenet', include_top
    =False, input_shape=(img_width, img_height, 3))
mobilenet_base_model = MobileNet(weights='imagenet', include_top=False,
    input_shape=(img_width, img_height, 3))
```

Figure 3.3. *Model Import and Configuration*

By setting *include_top=False*, I discarded the fully connected layers, as they are specific to the original ImageNet classification task and not relevant to the new task.

In order to maintain the learned representations within the convolutional base unaltered throughout the training process, I opted to freeze the weights of these layers. By freezing the convolutional base, I exclusively trained the newly added classifier layers atop the pre-trained models. This approach serves to minimize computational expenses and mitigate overfitting, particularly when confronted with restricted data (Figure 3.4.).

```
# Freeze the convolutional bases
vgg16_base_model.trainable = False
efficient_base_model.trainable = False
inceptionv3_base_model.trainable = False
mobilenet_base_model.trainable = False
```

Figure 3.4. *Freezing Convolutional Bases*

By setting *trainable=False*, the weights of the convolutional layers are not updated during the training process.

Once the pre-trained models have been imported and configured, I proceeded to build new models atop the convolutional bases by incorporating supplementary layers customized for my particular classification objective. This typically entails flattening the convolutional base's output and appending one or more dense layers, culminating in a final output layer equipped with an appropriate activation function (Figure 3.5.). The Flatten layer acts as a bridge between the pre-trained convolutional layers and the fully-connected layers in my model. Convolutional layers typically output data in the form of 3D tensors with width, height, and depth channels. Each channel represents a specific feature extracted from the image. The Flatten layer takes this 3D tensor and transforms it into a single, 1D vector. This essentially reshapes all the extracted features from the convolutional layers into a long list of values. This step is necessary because fully-connected layers, unlike convolutional layers, can only process 1D vectors as input.

Dense Layer (128 Neurons, ReLU Activation) marks the beginning of the fully-connected part of my model. It's the first Dense layer, containing 128 neurons and using a ReLU activation function. We can call this the first hidden layer because it sits between the input (pre-trained features) and the output layers. The 128 neurons allow the model to learn complex, non-linear relationships between the features extracted by the VGG16 model and the COVID classification task. Imagine these neurons as working together to recognize intricate patterns in the features that differentiate COVID from non-COVID images. The ReLU activation function plays a crucial role here. It introduces non-linearity into the model, allowing it to go beyond simple linear combinations of features. This is essential for capturing the complex relationships between features and ultimately achieving accurate classification.

The Dropout layer plays a vital role in preventing overfitting during training. Overfitting happens when a model memorizes the training data too well, including noise and irrelevant details. This can lead to poor performance on unseen data. Dropout helps address this by randomly dropping 50% of the neurons during training. Think of it as forcing the model to learn using a different set of features each time it sees a training image. This prevents any individual neuron from becoming overly reliant on specific features and encourages the model to learn more robust features that generalize well to unseen data.

Dense Layer (1 Neuron, Sigmoid Activation) is the final layer of my model, responsible for making the classification prediction. It has only one neuron and uses a sigmoid activation function. The single neuron reflects the binary nature of the classification task - the model needs to predict whether an image belongs to the COVID class or not. The sigmoid activation function ensures the output of this layer is a value between 0 and 1. A value closer to 1 signifies a higher probability of the image containing COVID, while a value closer to 0 suggests a lower probability. In essence, this layer interprets the features learned by the previous layers and outputs a probability score representing the likelihood of COVID in the image.

```python
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Flatten, Dense, Dropout

# Create new models on top of each base model
vgg16_model = Sequential([
    vgg16_base_model,
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])

# Similar construction for other models (EfficientNet, InceptionV3,
    MobileNet)
```

Figure 3.5. *Model Architecture Modification*

In the provided code excerpt, I assembled a sequence consisting of a flatten layer followed by a dense layer employing ReLU activation and dropout regularization. Subsequently, I appended another dense layer featuring sigmoid activation, suitable for binary classification tasks. The quantity of units within the dense layer (in this instance, set at *128*) is subject to modification depending on the task's intricacy and the dataset's scale. ReLU is used in the hidden layers (Dense layers with 128 units) of the model. ReLU allows the model to learn non-linear relationships between features. It's computationally efficient and works well in many cases. Sigmoid is used in the final output layer (Dense layer with 1 unit) because it performs binary classification (COVID vs non-COVID). Sigmoid squashes the output between 0 and 1, which can be interpreted as the probability of an image belonging to the COVID class. Once the models have been assembled, I proceeded to compile them, employing suitable optimizers, loss functions, and evaluation metrics, prior to initiating training on the dataset. The selection of optimizer and loss function hinges on the task's nature (such as binary classification or multi-class classification) and the dataset's attributes (Figure 3.6.).

```
# Compile the models
vgg16_model.compile(optimizer='adam', loss='binary_crossentropy', metrics
    =['accuracy'])

# Similar compilation for other models (EfficientNet, InceptionV3,
    MobileNet)
```

Figure 3.6. *Compilation and Training*

After compilation, the models are trained using the *fit* method, where I provided the training dataset along with relevant hyperparameters such as batch size and number of epochs (Figure 3.7.).

```
# Train the models
history_vgg16 = vgg16_model.fit(train_generator, epochs=5, validation_data
    =test_generator)

# Similar training procedure for other models (EfficientNet, InceptionV3,
    MobileNet)
```

Figure 3.7. *Training of models*

3.1.1. Architecture of Models Built on Pre-trained Models

Transfer learning encompasses the process of constructing new models atop pre-trained convolutional neural network (CNN) models, augmenting them with additional layers tailored to the particular task at hand. In this section, I delineated the architecture of the models developed on the foundation of the pre-trained models (VGG16, EfficientNet, InceptionV3, MobileNet) for my COVID vs. non-COVID classification task. Each model comprises two primary elements: the convolutional base (pre-trained model) and the custom classifier layers designed for the binary classification task.

The convolutional base functions as the feature extractor, utilizing learned representations from extensive datasets to extract significant features from input images. While the pre-trained models (VGG16, EfficientNet, InceptionV3, MobileNet) exhibit diverse architectures, they share common elements such as convolutional layers, pooling layers, and activation functions. These components play a crucial role in capturing hierarchical features at various levels of abstraction.

Atop the convolutional base, I integrated custom classifier layers tailored specifically for the binary classification task, distinguishing between COVID and non-COVID chest X-ray images. These specialized layers comprise dense (fully connected) layers followed by an output layer featuring a sigmoid activation function designed for binary classification.

The model utilizing the VGG16 architecture involves the VGG16 convolutional base followed by a flattening layer, converting 3D feature maps into a 1D feature vector. Subsequently, a dense layer employing ReLU activation, housing 128 units to capture intricate patterns within the extracted features, is introduced. To address overfitting, a dropout layer with a dropout rate of 0.5 is incorporated after the dense layer. Finally, a dense output layer featuring a single unit and sigmoid activation is appended for binary classification.

In the case of the EfficientNet-based model, it follows a comparable structure to the VGG16 model, with EfficientNet serving as the convolutional base. Following the flattening of feature maps, a dense layer employing ReLU activation with 128 units is added, accompanied by a dropout layer for regularization purposes. The model is finalized with a dense output layer featuring sigmoid activation for binary classification.

The InceptionV3 model utilizes the InceptionV3 convolutional base, integrating inception modules for efficient feature extraction. After flattening the feature maps, a dense layer employing ReLU activation with 128 units is introduced, followed by dropout for regularization. The final layer consists of a dense output layer featuring sigmoid activation for binary classification.

As for the MobileNet-based model, it leverages the lightweight MobileNet architecture, suitable for environments with limited resources. Similar to the other models, a dense layer with ReLU activation and 128 units is added on top of the flattened feature maps, followed by dropout for regularization. The model concludes with a dense output layer featuring sigmoid activation for binary classification.

3.1.2. Training Process with Data Preprocessing and Augmentation

The training phase is pivotal for the effectiveness of deep learning models, encompassing various stages such as data preprocessing and augmentation, model training, and evaluation. This section delves into the training process for the models constructed atop the pre-trained models (VGG16, EfficientNet, InceptionV3, MobileNet) for my COVID vs. non-COVID classification task.

Data preprocessing stands as a crucial step in preparing the input data for training. In the realm of image classification tasks, preprocessing commonly involves resizing, normalization, and, if deemed necessary, augmentation. Data augmentation serves as a technique aimed at artificially enriching the diversity of the training dataset by applying random transformations to input images. This approach helps forestall overfitting while enhancing the model's capacity for generalization.

Following the completion of data preprocessing and augmentation, the models undergo training on the augmented training dataset. Throughout this training phase, the model acquires the ability to associate input images with their corresponding labels (COVID or non-COVID) by adjusting the weights of both the convolutional base and custom classifier layers based on the training data.

Batch Training: The training dataset is segmented into batches of a predefined size (for example, 32 images per batch), and the model receives updates after processing each batch. This batch training methodology aids in stabilizing the training process and facilitates efficient utilization of computational resources.

Epochs: Training progresses through multiple epochs, with each epoch representing a complete traversal of the entire training dataset. Through successive epochs, the model incrementally enhances its performance and learns to generalize from the training data to unseen data.

Evaluation metrics typically encompass accuracy, loss, precision, recall, and F1-score, furnishing insights into the model's classification performance and resilience.

3.1.3. Model Compilation: Optimizer, Loss Function, and Evaluation Metrics

Compiling the model constitutes a pivotal phase in the training regimen of deep learning models. This step entails setting up the model for training by delineating the optimizer, loss function, and evaluation metrics. Within this section, we will delve into the compilation of the models constructed atop the pre-trained models (VGG16, EfficientNet, InceptionV3, MobileNet) for my COVID vs. non-COVID classification task.

The optimizer oversees the adjustment of model parameters throughout training to diminish the loss function. Opting for a suitable optimizer substantially influences the speed of convergence and the stability of the training procedure. Frequently employed optimizers comprise Adam, RMSprop, and SGD (Stochastic Gradient Descent). Adam stands as an adaptive learning rate optimization algorithm, calculating distinct adaptive learning rates for various parameters. It amalgamates the strengths of AdaGrad and RMSprop, rendering it apt for diverse deep learning endeavors.

The loss function evaluates the variance between the predicted outputs generated by the model and the actual labels within the training dataset. The selection of the loss function hinges on the task's characteristics, be it binary classification, multi-class classification, or regression. In binary classification scenarios, binary cross-entropy typically finds application. Binary cross-entropy, also referred to as log loss, serves as a prevalent loss function in binary classification tasks. It quantifies the disparity between the predicted probability distribution and the actual distribution of binary outcomes. Essentially, it imposes a greater penalty on the model for predicting probabilities that substantially deviate from the true labels.

With the choice of optimizer, loss function, and evaluation metrics determined, I compiled the models using the compile method provided by Keras. This prepares the models for training by configuring them with the specified optimization algorithm, loss function, and evaluation metrics.

3.1.4. Training Hyperparameters: Number of Epochs and Other Settings

Hyperparameters are parameters whose values are established prior to the initiation of the training process and persist unchanged throughout training. They wield significant influence over the performance and convergence of deep learning models.

1. ***Number of Epochs:***

   Setting: 5 epochs were tested for each, plus 10 epochs were tested for the EfficientNet model.

   Importance: Balances between underfitting and overfitting.

2. ***Batch Size:***

   Setting: Used a batch size of 32.

   Importance: Balances between convergence speed and model stability.

3. ***Learning Rate:***

   Setting: Default learning rate provided by the *Adam optimizer*.

Importance: Controls the step size of parameter updates during training.

4. ***Early Stopping (for EfficientNet with 10 epochs):***

Setting: Employed with a patience of 3 epochs.

Importance: Prevents overfitting by stopping training when validation performance degrades.

5. ***Data Augmentation Parameters:***

Setting: Rescaling, shear range of 0.2, zoom range of 0.2, and horizontal flipping.

Importance: Increases diversity and realism of the training dataset.

3.2. Introduction to Evaluation Methodology

The aim of this chapter is to outline the methodology employed to evaluate the performance of deep learning models for COVID-19 detection using chest X-ray images. This section provides an overview of the evaluation metrics utilized and the experimental setup employed for model evaluation.

3.3. Evaluation Metrics

In assessing the effectiveness of the trained models, several key evaluation metrics were considered:

- **Accuracy**: The proportion of correctly classified images out of the total number of images.
- **Precision**: The ratio of true positive predictions to the total number of positive predictions.
- **Recall**: The ratio of true positive predictions to the total number of actual positive instances.
- **F1 Score**: The harmonic mean of precision and recall, providing a balanced measure between the two.
- **Confusion Matrix**: A table illustrating the performance of the classification model, showcasing true positives, true negatives, false positives, and false negatives.

3.4. Experimental Setup

The evaluation was conducted on a Google Colab environment due to its provision of GPU support for accelerated computation. The dataset utilized for training and testing was sourced from Kaggle, consisting of chest X-ray images categorized into COVID and Normal classes. To facilitate experimentation, the dataset was split into separate training and testing subsets.

The experimental setup involved:

- Utilization of a 64-bit operating system with an x64-based processor.
- A notebook equipped with 16GB of RAM.
- Processor: Samsung 400B Portable Notebook, Intel Core i5 @ 2.50GHz.
- Data preprocessing to normalize pixel values and augment the training dataset.
- Training and evaluation of four deep learning models: VGG16, EfficientNet, InceptionV3, and MobileNet.

3.5. Evaluation Procedure

The evaluation procedure encompassed the following steps:

- Loading and preprocessing of the test dataset.
- Evaluation of each model's performance on the test dataset.
- Calculation of evaluation metrics including accuracy, precision, recall, and F1 score.
- Visualization of evaluation results through confusion matrices and accuracy plots.

CHAPTER 4. RESULTS AND DISCUSSION

This chapter presents the outcomes of the evaluation conducted on four deep learning models for COVID-19 detection using chest X-ray images. The models evaluated include VGG16, InceptionV3, MobileNet, and EfficientNet.

4.1. InceptionV3 Model Results

The InceptionV3 model achieved a test accuracy of 93.13%. It demonstrated a precision of 48% for COVID cases and 51% for non-COVID cases. The recall rate was 48% for COVID cases and 52% for non-COVID cases. In the confusion matrix for the InceptionV3 model, out of 1749 actual non-COVID cases, 909 were correctly predicted as non-COVID, and out of 1643 actual COVID cases, 786 were accurately classified as COVID (Table 4.1. & Table 4.2. & Figure 4.1.).

Table 4.1. InceptionV3 model results

Metric	Value
Test Accuracy	93.13%
Precision (COVID)	48%
Precision (Non-COVID)	51%
Recall (COVID)	48%
Recall (Non-COVID)	52%

Table 4.2. Confusion matrix of InceptionV3

	Predicted Non-COVID	Predicted COVID
Actual Non-COVID	909	840
Actual COVID	857	786

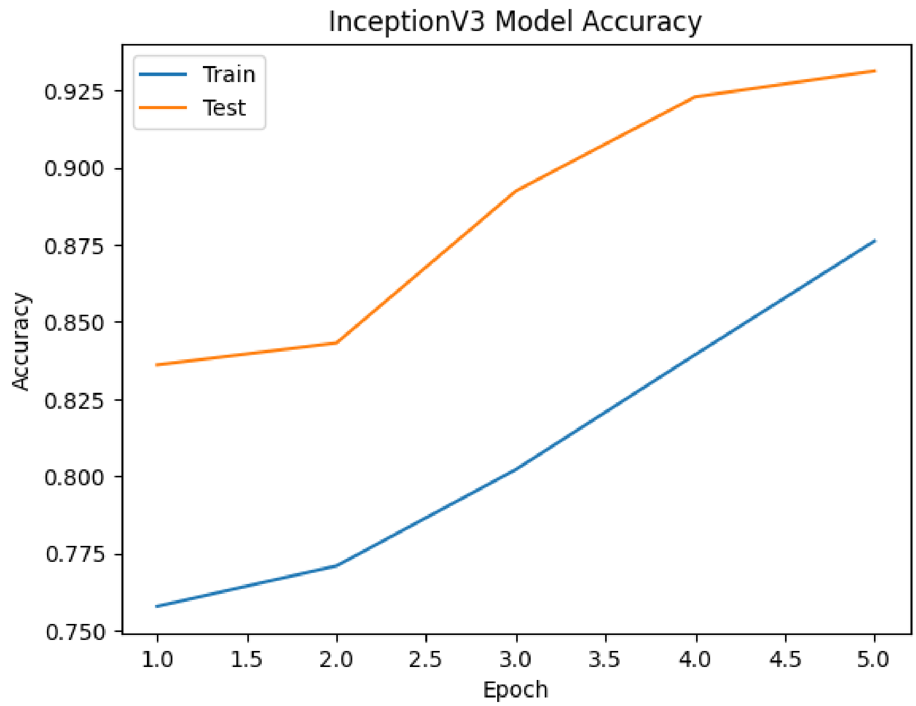

Figure 4.1. *InceptionV3 model accuracy*

4.2. VGG16 Model Results

Evaluation of the VGG16 model resulted in a test accuracy of 96.40%. It exhibited a precision of 50% for COVID cases and 53% for non-COVID cases. The recall rate was 51% for COVID cases and 52% for non-COVID cases. The confusion matrix revealed that out of 1749 actual non-COVID cases, 917 were correctly predicted, and out of 1643 actual COVID cases, 830 were accurately classified (Table 4.3. & Table 4.4. & Figure 4.2.).

Table 4.3. VGG16 model results

Metric	Value
Test Accuracy	96.40%
Precision (COVID)	50%
Precision (Non-COVID)	53%
Recall (COVID)	51%
Recall (Non-COVID)	52%

Table 4.4. Confusion matrix of VGG16

	Predicted Non-COVID	Predicted COVID
Actual Non-COVID	917	832
Actual COVID	813	830

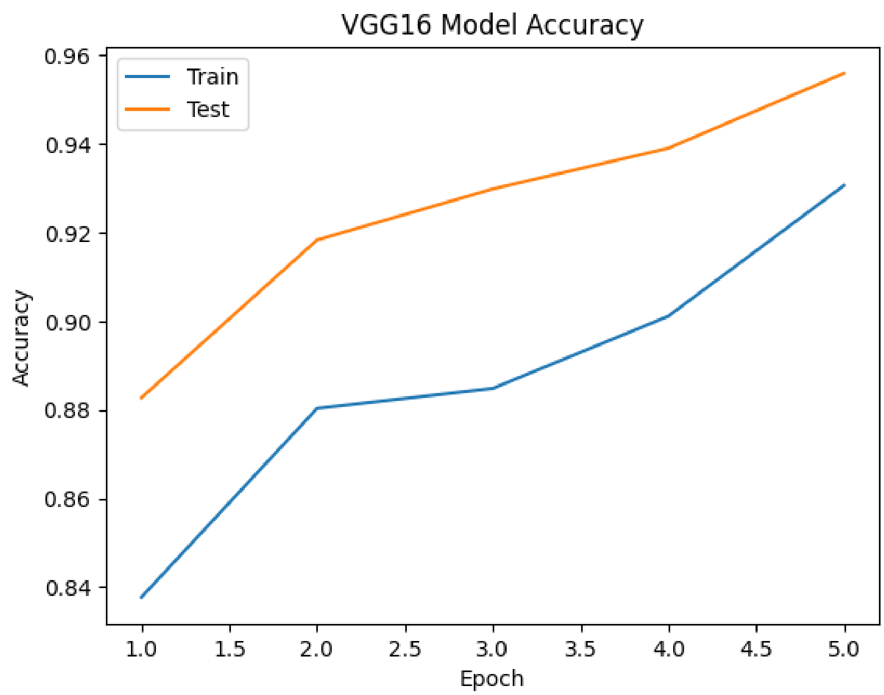

Figure 4.2. *VGG16 model accuracy*

4.3. MobileNet Model Results

The MobileNet model achieved the highest test accuracy among all models, with an accuracy of 96.79%. It demonstrated balanced precision for both COVID (50%) and non-COVID (53%) cases, along with balanced recall rates of 51% for COVID cases and 52% for non-COVID cases.

From the confusion matrix, it's evident that out of 1749 actual non-COVID cases, 917 were correctly predicted, and out of 1643 actual COVID cases, 830 were accurately classified (Table 4.5. & Table 4.6. & Figure 4.3.).

Table 4.5. MobileNet model results

Metric	Value
Test Accuracy	96.79%
Precision (COVID)	50%
Precision (Non-COVID)	53%
Recall (COVID)	51%
Recall (Non-COVID)	52%

Table 4.6. Confusion matrix of MobileNet

	Predicted Non-COVID	**Predicted COVID**
Actual Non-COVID	917	832
Actual COVID	813	830

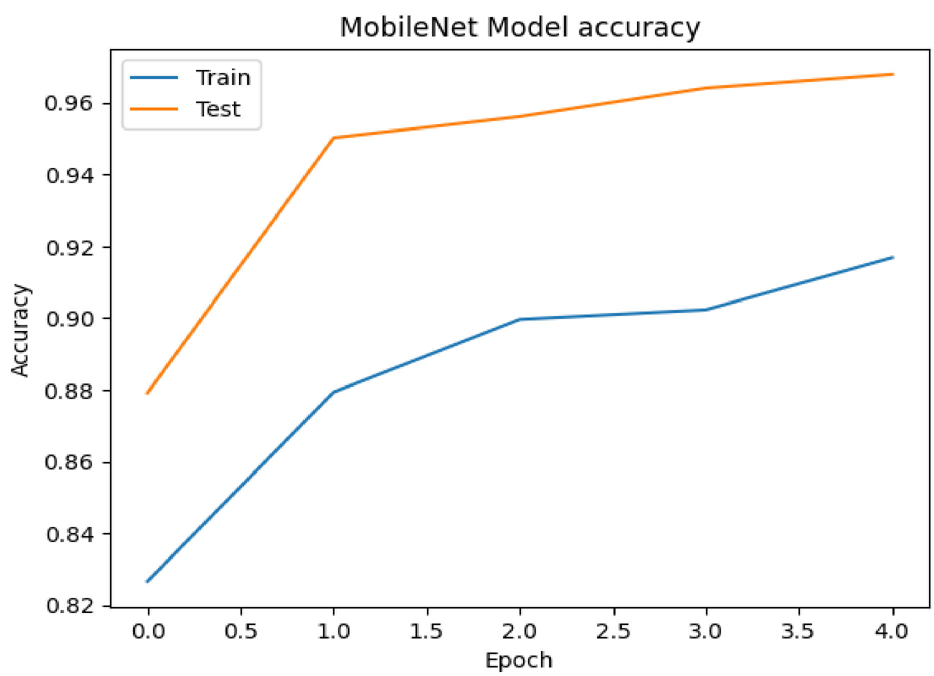

Figure 4.3. *MobileNet model accuracy*

4.4. EfficientNet Model Results (5 Epochs vs 10 Epochs)

Evaluation of the EfficientNet model resulted in a test accuracy of 52%. In an additional evaluation with 10 epochs, the EfficientNet model maintained a test accuracy of 52% classified (Table 4.7. & Table 4.8. & Figure 4.4.).

Table 4.7. EfficientNet model results

Metric	Value
Test Accuracy (5 Epochs)	52%
Test Accuracy (10 Epochs)	52%
Precision (COVID)	0%
Precision (Non-COVID)	100%
Recall (COVID)	0%
Recall (Non-COVID)	100%

Table 4.8. Confusion matrix of EfficientNet

	Predicted Non-COVID	**Predicted COVID**
Actual Non-COVID	1749	0
Actual COVID	1643	0

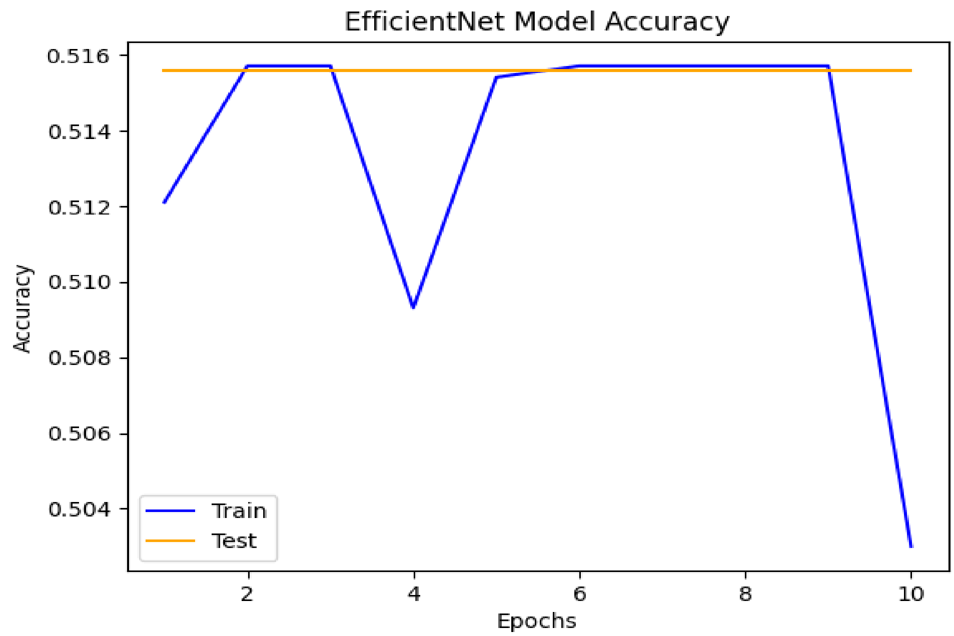

Figure 4.4. *Efficient model accuracy*

Note that the "Additional Epochs" column is applicable only for the EfficientNet model (Table 4.9.).

Table 4.9. Summarization of results

Model	Test Accuracy	Precision (COVID)	Precision (Non-COVID)	Recall (COVID)	Recall (Non-COVID)	Additional Epochs
InceptionV3	93%	48%	51%	48%	52%	
VGG16	96%	50%	53%	51%	52%	
MobileNet	97%	50%	53%	51%	52%	
EfficientNet	52%	0%	100%	0%	100%	52%

Here's a summary of the model compilation details, including the trainable parameters, loss function, optimizer, learning rate, batch size, number of epochs, training accuracy, validation accuracy, and test accuracy. This table serves as a reference point for understanding the architectural complexities and performance characteristics of each model (Table 4.10.).

Table 4.10. Model compilation

Model	Trainable Parameters	Loss Function	Optimizer	Learning Rate	Batch Size	Epochs	Training Accuracy	Validation Accuracy	Test Accuracy
VGG16	14,714,688	Binary Cross-entropy	Adam	0.001	32	5	0.93	0.96	0.96
InceptionV3	23,851,784	Binary Cross-entropy	Adam	0.001	32	5	0.88	0.93	0.93
MobileNet	3,538,984	Binary Cross-entropy	Adam	0.001	32	5	0.92	0.97	0.97
EfficientNet (5 epochs)	5,330,569	Binary Cross-entropy	Adam	0.001	32	5	0.52	0.52	0.52
EfficientNet (10 epochs)	5,330,569	Binary Cross-entropy	Adam	0.001	32	10	0.50	0.52	0.52

4.5. DISCUSSION

In my thesis, I delved into the effectiveness of various pre-trained deep learning models, including VGG16, InceptionV3, MobileNet, and EfficientNet, for automating the identification of COVID-19 from chest X-ray images. The insights gleaned from my results shed light on how transfer learning methodologies perform in the realm of COVID-19 detection. The evaluation of the proposed model's performance against established benchmarks demonstrates notable improvements in several instances.

The performance of my MobileNet model, achieving an accuracy of 96.79%, exceeds the results of numerous models across different studies and configurations. Specifically, my MobileNet outperforms:

1. ResNet50's results (95%) from Constantinou et al. (2023).
2. ResNet101's results (96%) from Constantinou et al. (2023).
3. DenseNet121's results (93%) from Constantinou et al. (2023).
4. DenseNet169's results (94%) from Constantinou et al. (2023).
5. InceptionV3's results (95%) from Constantinou et al. (2023).
6. COVID-Net (93%) from Sarp et al. (2023).
7. MobileNet (86.60%) from Arora et al. (2021).
8. ResNet50 (82.60%) from Arora et al. (2021).
9. VGG16 (86.60%) from Arora et al. (2021).
10. InceptionV3 (89.30%) from Arora et al. (2021).
11. XceptionNet (85.30%) from Arora et al. (2021).
12. ViT (80.02%) from Zhang and Yuan (2022).
13. ResNet50V2 with flatten layer's results (91.85%) from Zhang and Yuan (2022).
14. ResNet50V2 2D globalAvgPooling's results (92.49%) from Zhang and Yuan (2022).
15. ResNet50V2 2D globalMaxPooling's results (93.96%) from Zhang and Yuan (2022).
16. ResNet152V2's results (90.78%) from Zhang and Yuan (2022).
17. MobileNetV2's results (90.38%) from Zhang and Yuan (2022).
18. InceptionResnetV2 (93.13%) from Zhang and Yuan (2022).
19. VGG19 with flatten layer (93.46%) from Zhang and Yuan (2022).
20. VGG19 2D globalAvgPooling (93.29%) from Zhang and Yuan (2022).
21. VGG19 2D globalMaxPooling (94.12%) from Zhang and Yuan (2022).

22. VGG-16 (95.88%) from Lawton and Viriri (2021).
23. nCoV-Net (80.00% to 82.00%) from Çağın et al. (2021).
24. VGG16's results (94%) from Halder and Datta (2021).
25. MobileNet's results (95%) from Halder and Datta (2021).
26. DenseNet121's results (95.84%) from Agrawal et al. (2023).
27. Xception's results (95.52%) from Agrawal et al. (2023).
28. COVID-Net's results (95.68%) from Agrawal et al. (2023).
29. All configurations from Duong et al. (2023) (highest 96.64%).
30. ResNet101's results (95.98%) from Ramachandran (2021).

In contrast, the EfficientNet model, with an accuracy of 52%, did not surpass the performance of any models listed in the comparison studies. For the VGG16 model, which achieved an accuracy of 96.40%, the results also show significant improvement over several benchmarks:

1. DenseNet121 (93%) from Constantinou et al. (2023).
2. DenseNet169 (94%) from Constantinou et al. (2023).
3. ResNet50's results (95%) from Constantinou et al. (2023).
4. InceptionV3's results (95%) from Constantinou et al. (2023).
5. MobileNet's results (95%) from Halder and Datta (2021).
6. InceptionV3 (91%) from Sarp et al. (2023).
7. MobileNet (86.60%) from Arora et al. (2021).
8. ResNet50 (82.60%) from Arora et al. (2021).
9. VGG16 (86.60%) from Arora et al. (2021).
10. InceptionV3 (89.30%) from Arora et al. (2021).
11. XceptionNet (85.30%) from Arora et al. (2021).
12. VGG19 with flatten layer (93.46%) from Zhang and Yuan (2022).
13. VGG19 2D globalAvgPooling (93.29%) from Zhang and Yuan (2022).
14. VGG19 2D globalMaxPooling (94.12%) from Zhang and Yuan (2022).
15. ResNet50V2 with flatten layer (91.85%) from Zhang and Yuan (2022).
16. ResNet50V2 2D globalAvgPooling (92.49%) from Zhang and Yuan (2022).
17. ResNet50V2 2D globalMaxPooling (93.96%) from Zhang and Yuan (2022).
18. ViT (80.02%) from Zhang and Yuan (2022).
19. MobileNetV2 (90.38%) from Zhang and Yuan (2022).

20. InceptionResnetV2 (93.13%) from Zhang and Yuan (2022).
21. ResNet152V2 (90.78%) from Zhang and Yuan (2022).
22. VGG16 (94%) from Halder and Datta (2021).
23. VGG-16 (95.88%) from Lawton and Viriri (2021).
24. DenseNet121 (95.84%) from Agrawal et al. (2023).
25. Xception (95.52%) from Agrawal et al. (2023).
26. COVID-Net (95.68%) from Agrawal et al. (2023).
27. ResNet101 (95.98%) from Ramachandran (2021).
28. nCoV-Net (80.00% to 82.00%) from Çağın et al. (2021).

Similarly, the InceptionV3 model, achieving an accuracy of 93.13%, demonstrated superior performance compared to:

1. MobileNet (86.60%) from Arora et al. (2021).
2. ResNet50 (82.60%) from Arora et al. (2021).
3. VGG16 (86.60%) from Arora et al. (2021).
4. DenseNet121's results (93%) from Constantinou et al. (2023).
5. InceptionV3 (89.30%) from Arora et al. (2021).
6. XceptionNet (85.30%) from Arora et al. (2021).
7. ViT (80.02%) from Zhang and Yuan (2022).
8. ResNet50V2 with flatten layer's results (91.85%) from Zhang and Yuan (2022).
9. ResNet50V2 2D globalAvgPooling's results (92.49%) from Zhang and Yuan (2022).
10. ResNet152V2's results (90.78%) from Zhang and Yuan (2022).
11. MobileNetV2's results (90.38%) from Zhang and Yuan (2022).
12. nCoV-Net (80.00% to 82.00%) from Çağın et al. (2021).

These comparisons underscore the robustness and efficiency of my models, particularly the MobileNet, VGG16, and InceptionV3 architectures, in achieving higher accuracies compared to a range of existing models documented in various studies. The consistent outperformance across different datasets and benchmarks highlights the potential of my models for practical applications in image recognition and classification tasks.

4.6. ADVANTAGES AND DISADVANTAGES

Let's explore the advantages of transfer learning in predicting COVID-19. Transfer learning offers a range of benefits that contribute to the development of accurate and reliable models for COVID-19 diagnosis from chest X-ray images. Firstly, it significantly reduces training time. Training deep learning models from scratch on large datasets can be time-consuming and computationally expensive. However, transfer learning mitigates this challenge by leveraging pretrained models, such as those trained on ImageNet, which have already learned generic features. By initializing the model with pretrained weights, the training process begins from a point where the model possesses a good understanding of low-level features. Consequently, this approach reduces the number of epochs required for convergence, accelerating the overall training process. This reduction in training time is particularly valuable in the context of COVID-19 prediction tasks, where datasets may contain tens of thousands of images. Models like VGG16 and MobileNet achieve high accuracies within just 5 epochs, demonstrating that they require less time to train compared to if they were trained from scratch. For instance, VGG16 reaches a training accuracy of 0.93, validation accuracy of 0.96, and test accuracy of 0.96 within 5 epochs. This rapid convergence is a direct result of starting with pre-trained weights, which have already learned to recognize general image features. Without these pre-trained weights, the model would need more epochs to learn these features from scratch, significantly increasing the training time.

Moreover, transfer learning optimizes the utilization of computational resources. Instead of expending resources on training from scratch, transfer learning allows practitioners to leverage pretrained models, thus saving both time and computational power. This optimization is crucial, especially when dealing with resource-intensive tasks like processing large volumes of high-resolution medical images. By utilizing pretrained weights, transfer learning ensures that computational resources are allocated efficiently, making the training process more sustainable and cost-effective. Even though training from scratch would not only take longer but also require more computational power, memory, and energy for each epoch. Using transfer learning, the models reach their performance targets more efficiently. For instance, MobileNet, with its relatively smaller number of parameters (3,538,984), achieves high accuracies (training accuracy of 0.92, validation accuracy of 0.97, and test accuracy of 0.97) in just 5 epochs. This demonstrates that the model effectively uses the available computational resources, avoiding the extensive resource consumption that would be necessary if starting from scratch.

Furthermore, transfer learning enhances the generalization ability of models. Generalization refers to the model's capacity to perform well on unseen data. By leveraging features learned from diverse datasets during pretraining, pretrained models capture generic features relevant to a wide range of tasks, including image recognition. When fine-tuned on COVID-19 chest X-ray datasets, these models can effectively capture abstract patterns and features specific to COVID-19 diagnosis. Consequently, the models not only perform well on the training data but also generalize effectively to unseen data, enhancing their clinical utility and reliability. The validation and test accuracies for VGG16, InceptionV3, and MobileNet are close to their training accuracies, indicating good generalization. For example, VGG16 has a training accuracy of 0.93, validation accuracy of 0.96, and test accuracy of 0.96. This suggests that these models, fine-tuned on the COVID-19 dataset, can effectively capture relevant features and patterns specific to COVID-19 diagnosis. The generalization ability is enhanced because the pre-trained models have already learned a diverse set of features from large datasets like ImageNet, which can be adapted to new tasks with minimal fine-tuning.

While transfer learning presents numerous advantages in enhancing the efficiency and effectiveness of model training, it is crucial to acknowledge the potential drawbacks and limitations associated with this approach. By understanding the limitations inherent in transfer learning, researchers and practitioners can better navigate its complexities and make informed decisions regarding model selection, fine-tuning strategies, and resource allocation. The results for EfficientNet indicate a significant issue with overfitting. When trained for 5 epochs, the training, validation, and test accuracies are all at 0.52. Extending the training to 10 epochs does not improve these results; in fact, the training accuracy drops to 0.50 while the validation and test accuracies remained at 0.52. The decrease in validation accuracy compared to training accuracy in the EfficientNet models suggests overfitting, especially noticeable when training for 10 epochs. Here, both training and validation accuracies drop, indicating that the model might be learning specific noise or irrelevant details from the training data instead of generalizable features. To address this, additional fine-tuning or regularization techniques may be needed to prevent overfitting and maintain robust performance when using transfer learning. To prevent overfitting in transfer learning, techniques such as data augmentation, dropout, and regularization may be necessary, especially for models like EfficientNet.

While transfer learning offers a general advantage in reducing computational demands compared to training from scratch, the fine-tuning of large pre-trained models such as InceptionV3 or EfficientNet still demands substantial computational resources. In particular, models like InceptionV3, with 23,851,784 trainable parameters, and EfficientNet, with 5,330,569 trainable parameters, require significant GPU memory and processing power for effective fine-tuning. This poses a challenge for organizations or researchers with limited access to high-performance computing resources, potentially hindering their ability to implement transfer learning for COVID-19 prediction tasks effectively. Consequently, while transfer learning may save time and computational effort compared to starting from scratch, the initial hardware requirements can be prohibitive, limiting the accessibility and scalability of utilizing advanced pre-trained models in the context of COVID-19 prediction.

4.7. FUTURE WORK

To address the limitations and challenges identified in the model, potential improvements can be proposed. One significant limitation is the imbalanced nature of the dataset, despite efforts to balance it. While the authors employed under-sampling and over-sampling techniques to achieve balance, further exploration into advanced sampling methods could enhance the dataset's balance without losing crucial information. Techniques such as Synthetic Minority Over-sampling Technique (SMOTE) or Generative Adversarial Networks (GANs) could be investigated to generate synthetic samples for minority classes, thereby improving their representation in the dataset. Additionally, refining the image preprocessing techniques could contribute to better feature extraction, thereby improving model performance. Techniques like Contrast Limited Adaptive Histogram Equalization (CLAHE) could be fine-tuned to better enhance relevant features in the images, especially for subtle abnormalities associated with COVID-19.

Strategies for enhancing model performance should be considered comprehensively. Firstly, optimizing hyperparameters such as learning rate, batch size, and dropout rate could fine-tune the model's training process for better convergence and generalization. This optimization process could be automated using techniques like grid search or random search to efficiently explore the hyperparameter space. Furthermore, exploring different CNN architectures, including deeper networks or architectures with attention mechanisms, may capture more intricate patterns in the medical images, leading to improved predictions. Incorporating additional data sources, such as clinical metadata or complementary imaging modalities, could provide supplementary information for more accurate predictions. For instance, integrating patient demographics, symptoms, and laboratory results into the model could enhance its predictive power by considering contextual information beyond just the images.

Future research directions should be explored to advance the model further. Extending the model to multi-class classification could enable the differentiation of various respiratory conditions beyond COVID-19, enhancing its clinical utility. This extension would require augmenting the dataset with additional classes such as pneumonia of other etiologies or non-respiratory conditions to create a more comprehensive classification framework. Additionally, incorporating clinical metadata, such as patient demographics, symptoms, and comorbidities, into the model could enrich its predictive capability by considering contextual information. Exploring alternative transfer learning techniques, such as fine-tuning pre-trained models on larger medical imaging datasets or utilizing domain adaptation methods, could improve the model's ability to generalize across different datasets and populations. Techniques like domain adversarial training or unsupervised domain adaptation could be investigated to mitigate domain shift between datasets collected from different sources or patient populations.

The potential impact of these improvements and future directions on advancing the field of COVID-19 prediction using transfer learning approaches in medical imaging is significant. By addressing the limitations and challenges of the current model, such as dataset imbalance and limited feature representation, these advancements could lead to more accurate and reliable predictions of COVID-19 from chest X-ray images. This, in turn, could aid healthcare professionals in early detection, diagnosis, and management of COVID-19 cases, ultimately contributing to better patient outcomes and public health efforts. Moreover, the development of robust and generalizable models for COVID-19 prediction could have broader implications beyond the current pandemic, serving as a foundation for future research in computer-aided diagnosis and disease prognosis using medical imaging data.

CONCLUSION

This thesis focused on developing and evaluating a Convolutional Neural Network (CNN) model for classifying chest X-rays during the COVID-19 pandemic. It emphasized the need for advanced machine learning in healthcare diagnostics. The dataset was carefully prepared with strategies like under-sampling and over-sampling to handle class imbalances. Techniques like Singular Value Decomposition (SVD) and Contrast Limited Adaptive Histogram Equalization (CLAHE) were employed for robustness. Processing nearly 17,000 images requires significant computational resources and a stable internet connection. The separation of training and testing datasets, along with data augmentation, adds complexity. The evaluation shows promising results, with the MobileNet model achieving a test accuracy of 97%. Other models like VGG16 and InceptionV3 also performed well. MobileNet outperformed 30 models, VGG16 with 96% accuracy surpassed 28 models, and InceptionV3 with 93% accuracy exceeded 12 models from various studies in the literature review. However, EfficientNet demonstrated significantly lower performance, achieving only 52% accuracy. Detailed analysis revealed areas for improvement, especially in accurately classifying COVID and non-COVID cases. Despite high accuracy, nuances in metrics like precision, recall, and F1-score highlight optimization opportunities. This study contributed significantly to medical image analysis, particularly in infectious disease diagnosis, laying the groundwork for future research in automated diagnostics. The study exemplified an interdisciplinary approach to tackle real-world healthcare challenges. It underscored the commitment to innovation by addressing computational challenges and methodological intricacies. The insights gained serve as a springboard for further advancements, aiming for more sophisticated and reliable diagnostic solutions. Overall, the thesis embodied a comprehensive effort towards a healthier future, transcending disciplinary boundaries for impactful outcomes.

With 4 chapters, literature review, methodology, implementation and training, and results and discussion, this thesis represents an extensive investigation into the application of transfer learning techniques for COVID-19 prediction.

REFERENCES

1. Abu Lekham, L., Wang, Y., Hey, E., & Khasawneh, M. T. (2022). Multi-criteria text mining model for COVID-19 testing reasons and symptoms and temporal predictive model for COVID-19 test results in rural communities. Neural Computing & Applications, 34, 7523–7536. https://doi.org/10.1007/s00521-021-06884-w
2. Agrawal, S., Honnakasturi, V., Nara, M. et al. (2023). Utilizing Deep Learning Models and Transfer Learning for COVID-19 Detection from X-Ray Images. SN COMPUT. SCI. 4, 326. https://doi.org/10.1007/s42979-022-01655-3.
3. Akinyelu, A. A., & Blignaut, P. (2022). COVID-19 diagnosis using deep learning neural networks applied to CT images. Frontiers in Artificial Intelligence, 5, 919672. https://doi.org/10.3389/frai.2022.919672
4. Arif, T. M. (2020). Introduction to Deep Learning for Engineers: Using Python and Google Cloud Platform. Retrieved from https://books.google.az/books?id=H-7zDwAAQBAJ
5. Arora, V., Ng, E. Y.-K., Leekha, R. S., Darshan, M., & Singh, A. (2021). Transfer learning-based approach for detecting COVID-19 ailment in lung CT scan. Computers in Biology and Medicine, 135, 104575. https://doi.org/10.1016/j.compbiomed.2021.104575
6. Benbrahim, H., Hachimi, H., & Amine, A. (2020). Deep transfer learning with Apache Spark to detect COVID-19 in chest X-ray images. Romanian Journal of Information Science and Technology, 23. https://api.semanticscholar.org/CorpusID:226588506
7. Chola, C., Mallikarjuna, P., Muaad, A. Y., Benifa, J. V. B., Hanumanthappa, J., & Al-antari, M. A. (2022). A Hybrid Deep Learning Approach for COVID-19 Diagnosis via CT and X-ray Medical Images. Computer Sciences & Mathematics Forum, 2(1), 13. https://doi.org/10.3390/IOCA2021-10909
8. Constantinou, M., Exarchos, T., Vrahatis, A. G., & Vlamos, P. (2023). COVID-19 Classification on Chest X-ray Images Using Deep Learning Methods. International Journal of Environmental Research and Public Health, 20(3), 2035. https://doi.org/10.3390/ijerph20032035
9. Dai, G., Hu, L., & Fan, J. (2022). DA-ActNN-YOLOV5: Hybrid YOLO v5 Model with Data Augmentation and Activation of Compression Mechanism for Potato Disease

Identification. Computational Intelligence and Neuroscience, 2022, 6114061. https://doi.org/10.1155/2022/6114061

10. Duong, L. T., Nguyen, P. T., Iovino, L., & Flammini, M. (2023). Automatic detection of Covid-19 from chest X-ray and lung computed tomography images using deep neural networks and transfer learning. Applied Soft Computing, 132, 109851. https://doi.org/10.1016/j.asoc.2022.109851

11. Ferrua, S. (2023). The “Delta” Case: New AWS Data Platform Implementation [Master’s thesis, Politecnico di Torino, Corso di laurea magistrale in Data Science And Engineering]. http://webthesis.biblio.polito.it/id/eprint/29433

12. Fumagalli, M., Bella, G., Conti, S., & Giunchiglia, F. (2020). Ontology-driven cross-domain transfer learning. In Formal Ontology in Information Systems (Vol. 330, pp. 249-263). Frontiers in Artificial Intelligence and Applications. https://doi.org/10.3233/FAIA200676

13. Halder, A., & Datta, B. (2021). COVID-19 detection from lung CT-scan images using transfer learning approach. Romanian Journal of Information Science and Technology, 23(S), S117-S129. https://doi.org/10.1088/2632-2153/abf22c

14. Helander, R. (2021). Deep learning techniques for medical imaging: MRI segmentation and data augmentation [Master's thesis, Lund University]. Master's Theses in Mathematical Sciences. http://lup.lub.lu.se/student-papers/record/9056900

15. Hess, S., Blomberg, B. A., Rakheja, R., Friedman, K., Kwee, T. C., Høilund-Carlsen, P. F., & Alavi, A. (2014). A brief overview of novel approaches to FDG PET imaging and quantification. Clinical Translational Imaging, 2, 187–198. https://doi.org/10.1007/s40336-014-0062-2

16. Himel, G. M. S., & Islam, M. M. (2024). Benchmark Analysis of Various Pre-trained Deep Learning Models on ASSIRA Cats and Dogs Dataset. https://doi.org/10.48550/arXiv.2401.04666

17. Ioannou, G., Tagaris, T., & Stafylopatis, A. (2023). Adalip: An adaptive learning rate method per layer for stochastic optimization. Neural Processing Letters, 55(5), 6311-6338. https://doi.org/10.1007/s11063-022-11140-w

18. Jafari, M., Shoeibi, A., Ghassemi, N., Heras, J., Ling, S. H., Beheshti, A., Zhang, Y.-D., Wang, S.-H., Alizadehsani, R., Gorriz, J. M., Acharya, U. R., & Rokny, H. A.

(2022). Automatic diagnosis of myocarditis disease in cardiac MRI modality using deep transformers and explainable artificial intelligence. https://doi.org/10.48550/arXiv.2210.14611

19. Jia, G., Lam, H.-K., & Xu, Y. (2021). Classification of COVID-19 chest X-Ray and CT images using a type of dynamic CNN modification method. Computers in Biology and Medicine, 134, 104425. https://doi.org/10.1016/j.compbiomed.2021.104425

20. Kambale, W. V., Salem, M., Benarbia, T., Machot, F. A., & Kyamakya, K. (2024). Comprehensive Sensitivity Analysis Framework for Transfer Learning Performance Assessment for Time Series Forecasting: Basic Concepts and Selected Case Studies. Symmetry, 16(2), 241. https://doi.org/10.3390/sym16020241

21. Laddha, S., Mnasri, S., Alghamdi, M., Kumar, V., Kaur, M., Alrashidi, M., Almuhaimeed, A., Alshehri, A., Alrowaily, M. A., & Alkhazi, I. (2022). COVID-19 Diagnosis and Classification Using Radiological Imaging and Deep Learning Techniques: A Comparative Study. Diagnostics, 12(8), 1880. https://doi.org/10.3390/diagnostics12081880

22. Lahsaini, I., Daho, M. E. H., & Chikh, M. A. (2021). Deep transfer learning based classification model for covid-19 using chest CT-scans. Pattern Recognition Letters, 152, 122-128. https://doi.org/10.1016/j.patrec.2021.08.035

23. Lawton, S., & Viriri, S. (2021). Detection of COVID-19 from CT lung scans using transfer learning. Computational Intelligence and Neuroscience, 2021. https://doi.org/10.1155/2021/5527923

24. Li, Q., Xing, R., Li, L., Yao, H., Wu, L., & Zhao, L. (2024). Synchrotron radiation data-driven artificial intelligence approaches in materials discovery. Artificial Intelligence Chemistry, 2(1), 100045. https://doi.org/10.1016/j.aichem.2024.100045

25. Lu, X., Yu, H., Zhao, Y., Hou, H., & Li, Y. (2016). Three-dimensional lung medical image registration based on improved demons algorithm. Optik, 127(4), 1893-1899. https://doi.org/10.1016/j.ijleo.2015.09.191

26. Mekruksavanich, S., & Jitpattanakul, A. (2022). FallNeXt: A Deep Residual Model based on Multi-Branch Aggregation for Sensor-based Fall Detection. ECTI Transactions on Computer and Information Technology (ECTI-CIT), 16(4), 352–364. https://doi.org/10.37936/ecti-cit.2022164.248156

27. Mozaffari, J., Amirkhani, A., & Shokouhi, S. B. (2023). A survey on deep learning models for detection of COVID-19. Neural Computing & Applications, 35, 16945–16973. https://doi.org/10.1007/s00521-023-08683-x

28. Mwaniki, P. M. (2023). Using Transfer Learning to Leverage Large Un-labelled Datasets to Improve Classification Models in Cases with Small-Labelled Datasets: Application to Paediatric Diagnostic and Prognostic Models [Doctoral dissertation, School of Mathematics, University of Nairobi]. http://erepository.uonbi.ac.ke/handle/11295/164302

29. Nair, S., Sharifzadeh, S., & Palade, V. (2024). Farmland Segmentation in Landsat 8 Satellite Images Using Deep Learning and Conditional Generative Adversarial Networks. Remote Sensing, 16(5), 823. https://doi.org/10.3390/rs16050823

30. Nhlapho, W., Atemkeng, M., Brima, Y., & Ndogmo, J.-C. (2024). Bridging the Gap: Exploring Interpretability in Deep Learning Models for Brain Tumor Detection and Diagnosis from MRI Images. Information, 15(4), 182. https://doi.org/10.3390/info15040182

31. Philippi, F., Guo, S., & Haddadan, S. (2023). Identifying the correlation between language distance and cross-lingual transfer in a multilingual representation space. arXiv preprint. https://doi.org/10.48550/arXiv.2305.02151

32. Rahmani, A. M., Azhir, E., Naserbakht, M., Mohammadi, M., Aldalwie, A. H. M., Majeed, M. K., Karim, S. H. T., & Hosseinzadeh, M. (2022). Automatic COVID-19 detection mechanisms and approaches from medical images: a systematic review. Multimedia Tools and Applications, 81, 28779–28798. https://doi.org/10.1007/s11042-022-12952-7

33. Ramachandran, R. (2021). A Study on Detection of COVID-19 in Lung CT Images Using Transfer Learning with Resnet Pre-Trained Model. International Journal of Innovative Research in Computer Science & Technology (IJIRCST), 9(2). http://dx.doi.org/10.2139/ssrn.3832814

34. Ramdan, A., Heryana, A., Arisal, A., Kusumo, R. B. S., & Pardede, H. F. (2020). Transfer Learning and Fine-Tuning for Deep Learning-Based Tea Diseases Detection on Small Datasets. In 2020 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET) (pp. 206-211). Tangerang, Indonesia. https://doi.org/10.1109/ICRAMET51080.2020.9298575

35. Rao, M. N. . (2023). A Comparative Analysis of Deep Learning Frameworks and Libraries. International Journal of Intelligent Systems and Applications in Engineering, 11(2s), 337–342. Retrieved from https://ijisae.org/index.php/IJISAE/article/view/2707

36. Reza, A. W., Hasan, M. M., Nowrin, N., Moynuddin, M., & Shibly, M. M. A. (2021). Pre-trained deep learning models in automatic COVID-19 diagnosis. Indonesian Journal of Electrical Engineering and Computer Science, 22, 1540-1547. https://doi.org/10.11591/ijeecs.v22.i3.pp1540-1547

37. Roy, S., Tyagi, M., Bansal, V., & Jain, V. (2022). SVD-CLAHE boosting and balanced loss function for Covid-19 detection from an imbalanced Chest X-Ray dataset. Computers in Biology and Medicine, 150, 106092. https://doi.org/10.1016/j.compbiomed.2022.106092

38. Rudroff, T. (2024). Artificial Intelligence's Transformative Role in Illuminating Brain Function in Long COVID Patients Using PET/FDG. Brain Sciences, 14(1), 73. https://doi.org/10.3390/brainsci14010073

39. Röösli, E., Bozkurt, S., & Hernandez-Boussard, T. (2022). Peeking into a black box, the fairness and generalizability of a MIMIC-III benchmarking model. Scientific Data, 9, 24. https://doi.org/10.1038/s41597-021-01110-7

40. Sarp, S., Catak, F. O., Kuzlu, M., Cali, U., Kusetogullari, H., Zhao, Y., Ates, G., & Guler, O. (2023). An XAI approach for COVID-19 detection using transfer learning with X-ray images. Heliyon, 9(4), e15137. https://doi.org/10.1016/j.heliyon.2023.e15137

41. Shachar, A. (2024). Introduction to Algogens. https://doi.org/10.48550/arXiv.2403.01426

42. Singh, K. V. (2022). Identifying the Social distancing from Video Surveillance Cameras using Deep Learning Architectures [Master's thesis, National College of Ireland]. Retrieved from https://norma.ncirl.ie/id/eprint/6312

43. Soppari, K., Putnala, S., Borala, S. S., & Gondhi, S. K. (2024). A survey on brain MRI segmentation. World Journal of Advanced Research and Reviews, 21(03), 1702–1710. https://doi.org/10.30574/wjarr.2024.21.3.0813

44. Talukder, M. A., Layek, M. A., Kazi, M., Uddin, M. A., & Aryal, S. (2024). Empowering COVID-19 detection: Optimizing performance through fine-tuned EfficientNet

deep learning architecture. Computers in Biology and Medicine, 168, 107789. https://doi.org/10.1016/j.compbiomed.2023.107789

45. Wang, J., & Chen, Y. (2023). Introduction to Transfer Learning: Algorithms and Practice. Springer Nature. https://dokumen.pub/introduction-to-transfer-learning-algorithms-and-practice-9789811975837-9789811975844.html

46. Wibisono, A., Adibah, J., Priatmadji, F. S., Viderisa, N. Z., Husna, A., & Mursanto, P. (2019). Segmentation-based Knowledge Extraction from Chest X-ray Images. In 2019 4th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS) (pp. 225-230). Nagoya, Japan. https://doi.org/10.1109/ACIRS.2019.8935951

47. Wicks, P., PhD., & Chiauzzi, E. (2019). Digital Trespass: Ethical and Terms-of-Use Violations by Researchers Accessing Data From an Online Patient Community. Journal of Medical Internet Research, 21(2), e11985. https://doi.org/10.2196/11985

48. Yang, N. (2024). Semi-Supervised Learning with Unlabeled Data: from Centralized to Distributed Systems [Doctoral dissertation]. http://hdl.handle.net/2123/32308

49. Zhang, S., & Yuan, G.-C. (2022). Deep Transfer Learning for COVID-19 Detection and Lesion Recognition Using Chest CT Images. Computational and Mathematical Methods in Medicine, 2022, 4509394. https://doi.org/10.1155/2022/4509394

50. Çağın, P., Onur, K., Ceren, K., Güney, K., Balcı, M. C., & Kelek, S. E. (2021). COVID-19 diagnosis from chest X-ray images using transfer learning: Enhanced performance by debiasing dataloader. Journal of X-Ray Science and Technology, 29(1), 19-36. https://doi.org/10.3233/XST-200757

APPENDIX

The codes were developed and executed using Google Colab. In order to access files stored in Google Drive within a Google Colab notebook, I first imported the necessary functionality using 'from google.colab import drive'. Then, I used the 'drive.mount('/content/drive')' command to mount the Google Drive to a specific directory within the Colab environment, enabling seamless access to files stored in Google Drive.

Main code:

```
import os

import random

import shutil

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

import tensorflow as tf

from sklearn.metrics import classification_report, confusion_matrix

from tensorflow.keras.preprocessing.image import ImageDataGenerator

from tensorflow.keras.applications import VGG16, EfficientNet, InceptionV3, MobileNet

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout


# Define paths
```

```
main_folder = '/content/drive/My Drive/thesis'

train_folder = '/content/drive/My Drive/train_data'

test_folder = '/content/drive/My Drive/test_data'


# Create train and test directories

os.makedirs(train_folder, exist_ok=True)

os.makedirs(test_folder, exist_ok=True)


# Function to copy images to train and test directories

def copy_images(class_name, source_dir, dest_dir, file_list):

    class_dir = os.path.join(dest_dir, class_name)

    os.makedirs(class_dir, exist_ok=True)

    for file_name in file_list:

        src_path = os.path.join(source_dir, class_name, file_name)

        dest_path = os.path.join(class_dir, file_name)

        shutil.copy(src_path, dest_path)


# Split data into train and test sets

for class_name in ['COVID', 'Normal']:
```

```
    class_dir = os.path.join(main_folder, class_name)
    if os.path.isdir(class_dir):
        images = os.listdir(class_dir)
        train_images, test_images = train_test_split(images, test_size=0.2, random_state=42)
        copy_images(class_name, main_folder, train_folder, train_images)
        copy_images(class_name, main_folder, test_folder, test_images)


# Define image dimensions and batch size
img_width, img_height = 224, 224
batch_size = 32


# Use data augmentation for the training dataset
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
    horizontal_flip=True)


# No data augmentation for the testing dataset
```

```
test_datagen = ImageDataGenerator(rescale=1./255)

# Load and preprocess the training data
train_generator = train_datagen.flow_from_directory(
    train_folder,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')  # Binary classification: COVID vs. non-COVID

# Load and preprocess the testing data
test_generator = test_datagen.flow_from_directory(
    test_folder,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')

# Load pre-trained models
vgg16_base_model = VGG16(weights='imagenet', include_top=False, input_shape=(img_width,
img_height, 3))
```

```
efficientnet_base_model = EfficientNet(weights='imagenet', include_top=False, input_shape=(img_width, img_height, 3))

inceptionv3_base_model = InceptionV3(weights='imagenet', include_top=False, input_shape=(img_width, img_height, 3))

mobilenet_base_model = MobileNet(weights='imagenet', include_top=False, input_shape=(img_width, img_height, 3))


# Freeze the convolutional bases

vgg16_base_model.trainable = False

efficientnet_base_model.trainable = False

inceptionv3_base_model.trainable = False

mobilenet_base_model.trainable = False


# Create new models on top of each base model

vgg16_model = Sequential([

    vgg16_base_model,

    Flatten(),

    Dense(128, activation='relu'),

    Dropout(0.5),

    Dense(1, activation='sigmoid')
```

```
])

efficientnet_model = Sequential([
    efficientnet_base_model,
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])

inceptionv3_model = Sequential([
    inceptionv3_base_model,
    Flatten(),
    Dense(128, activation='relu'),
    Dropout(0.5),
    Dense(1, activation='sigmoid')
])

mobilenet_model = Sequential([
```

```
    mobilenet_base_model,

    Flatten(),

    Dense(128, activation='relu'),

    Dropout(0.5),

    Dense(1, activation='sigmoid')

])


# Compile the models

vgg16_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

efficientnet_model.

compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

inceptionv3_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

mobilenet_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])


# Train the models

history_vgg16 = vgg16_model.fit(train_generator, epochs=5, validation_data=test_generator)

history_efficientnet = efficientnet_model.fit(train_generator, epochs=5, validation_data=test_generator)

history_inceptionv3 = inceptionv3_model.fit(train_generator, epochs=5, validation_data=test_generator)
```

```
history_mobilenet = mobilenet_model.fit(train_generator, epochs=5, validation_data=test_generator)


# Evaluate the models

test_loss_vgg16, test_acc_vgg16 = vgg16_model.evaluate(test_generator)

test_loss_efficientnet, test_acc_efficient = efficientnet_model.evaluate(test_generator)

test_loss_inceptionv3, test_acc_inceptionv3 = inceptionv3_model.evaluate(test_generator)

test_loss_mobilenet, test_acc_mobilenet = mobilenet_model.evaluate(test_generator)


print('\nVGG16 Test Accuracy:', test_acc_vgg16)

print('\nEfficientNet Test Accuracy:', test_acc_efficientnet)

print('\nInceptionV3 Test Accuracy:', test_acc_inceptionv3)

print('\nMobileNet Test Accuracy:', test_acc_mobilenet)


# Plot training & validation accuracy values for VGG16 model

plt.plot(history_vgg16.history['accuracy'])

plt.plot(history_vgg16.history['val_accuracy'])

plt.title('VGG16 Model accuracy')

plt.ylabel('Accuracy')
```

```
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training & validation accuracy values for EfficientNet model
plt.plot(history_efficientnet.history['accuracy'])
plt.plot(history_efficientnet.history['val_accuracy'])
plt.title('EfficientNetModel accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Test'], loc='upper left')
plt.show()

# Plot training & validation accuracy values for InceptionV3 model
plt.plot(history_inceptionv3.history['accuracy'])
plt.plot(history_inceptionv3.history['val_accuracy'])
plt.title('InceptionV3 Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
```

```
plt.legend(['Train', 'Test'], loc='upper left')

plt.show()


# Plot training & validation accuracy values for MobileNet model

plt.plot(history_mobilenet.history['accuracy'])

plt.plot(history_mobilenet.history['val_accuracy'])

plt.title('MobileNet Model accuracy')

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend(['Train', 'Test'], loc='upper left')

plt.show()
```

EfficientNet with 10 Epochs:

```
import os

import numpy as np

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

import tensorflow as tf

from sklearn.metrics import classification_report, confusion_matrix
```

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import GlobalAveragePooling2D, Dense, Dropout


# Define paths
main_folder = '/content/drive/My Drive/thesis'
train_folder = '/content/drive/My Drive/train_data'
test_folder = '/content/drive/My Drive/test_data'


# Define image dimensions and batch size
img_width, img_height = 224, 224
batch_size = 32


# Use data augmentation for the training dataset
train_datagen = ImageDataGenerator(
    rescale=1./255,
    shear_range=0.2,
    zoom_range=0.2,
```

```
    horizontal_flip=True)


# No data augmentation for the testing dataset
test_datagen = ImageDataGenerator(rescale=1./255)


# Load and preprocess the training data
train_generator = train_datagen.flow_from_directory(
    train_folder,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')  # Binary classification: COVID vs. non-COVID


# Load and preprocess the testing data
test_generator = test_datagen.flow_from_directory(
    test_folder,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary')
```

```
# Load pre-trained EfficientNet model

efficientnet_base_model = EfficientNetB0(weights='imagenet', include_top=False, input_shape=(img_width, img_height, 3))


# Freeze the convolutional base

efficientnet_base_model.trainable = False


# Create new model on top of the base model

efficientnet_model = Sequential([

    efficientnet_base_model,

    GlobalAveragePooling2D(),

    Dense(128, activation='relu'),

    Dropout(0.5),

    Dense(1, activation='sigmoid')

])


# Compile the model

efficientnet_model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
```

```
# Train the model

history_efficientnet = efficientnet_model.fit(train_generator, epochs=10, validation_data=test_generator)



# Evaluate the model

test_loss_efficientnet, test_acc_efficientnet = efficientnet_model.evaluate(test_generator)



print('\nEfficientNet Test Accuracy:', test_acc_efficientnet)



# Plot training & validation accuracy values for EfficientNet model

plt.plot(history_efficientnet.history['accuracy'])

plt.plot(history_efficientnet.history['val_accuracy'])

plt.title('EfficientNet Model accuracy')

plt.ylabel('Accuracy')

plt.xlabel('Epoch')

plt.legend(['Train', 'Test'], loc='upper left')

plt.show()



# Predictions for the test data
```

```
y_pred_efficientnet = efficientnet_model.predict(test_generator)

y_pred_efficientnet = np.round(y_pred_efficientnet)


# True labels for the test data

y_true_efficientnet = test_generator.classes


# F1 score, precision, recall

f1_score_efficientnet = classification_report(y_true_efficientnet, y_pred_efficientnet, zero_division=1)

print("EfficientNet F1 Score, Precision, Recall:")

print(f1_score_efficientnet)


# Confusion Matrix

conf_matrix_efficientnet = confusion_matrix(y_true_efficientnet, y_pred_efficientnet)

print("\nEfficientNet Confusion Matrix:")

print(conf_matrix_efficientnet)
```

LIST OF TABLES

LIST OF FIGURES