DETECTION OF COVID-19 PNEUMONIA ON COMPUTED TOMOGRAPHY IMAGES USING A LIGHTWEIGHT DEEP LEARNING MODEL

SERENA LOW WOAN CHING

FACULTY OF ENGINEERING
UNIVERSITI MALAYA
KUALA LUMPUR

2023

DETECTION OF COVID-19 PNEUMONIA ON COMPUTED TOMOGRAPHY IMAGES USING A LIGHTWEIGHT DEEP LEARNING MODEL

SERENA LOW WOAN CHING

THESIS SUBMITTED IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

FACULTY OF ENGINEERING
UNIVERSITI MALAYA
KUALA LUMPUR

2023

UNIVERSITI MALAYA

ORIGINAL LITERARY WORK DECLARATION

Name of Candidate: Serena Low Woan Ching

Matric No: 17058143/2

Name of Degree: Doctor of Philosophy

Title of Project Paper/Research Report/Dissertation/Thesis ("this Work"): Detection of COVID-19 Pneumonia on Computed Tomography Images Using A Lightweight Deep Learning Model

Field of Study: Signal and Systems, Department of Electrical Engineering

I do solemnly and sincerely declare that:

(1) I am the sole author/writer of this work;
(2) This work is original;
(3) Any use of any work in which copyright exists was done by way of fair dealing and for permitted purposes and any excerpt or extract from, or reference to or reproduction of any copyright work has been disclosed expressly and sufficiently and the title of the work and its authorship have been acknowledged in this work;
(4) I do not have any actual knowledge nor do I ought reasonably to know that the making of this work constitutes an infringement of any copyright work;
(5) I hereby assign all and every rights in the copyright to this work to the Universiti Malaya ("UM"), who henceforth shall be owner of the copyright in this work and that any reproduction or use in any form or by any means whatsoever is prohibited without the written consent of UM having been first had and obtained;
(6) I am fully aware that if in the course of making this work I have infringed any copyright whether intentionally or otherwise, I may be subject to legal action or any other action as may be determined by UM.

Candidate's Signature Date: 11/12/2023

Subscribed and solemnly declared before,

Witness's Signature Date: 11/12/2023

Name:

Designation:

DETECTION OF COVID-19 PNEUMONIA ON COMPUTED TOMOGRAPHY IMAGES USING A LIGHTWEIGHT DEEP LEARNING MODEL

ABSTRACT

SARS-CoV-2, also known as COVID-19, is a novel contagious respiratory disease discovered in 2019 that caused a worldwide pandemic that claimed many lives. The virus epidemic was initially discovered in December 2019 in Wuhan, China. It immediately escalated into an international crisis, causing widespread illness, death, and significant socio-economic disruptions. The rapid and accurate identification of COVID-19 is crucial to the ongoing global epidemic. One of the various medical devices is computed tomography (CT) imaging, which has shown promise in its application to detect distinctive patterns associated with lung tissue deterioration. The common practice is relying on radiologists to diagnose the CT images, which is time-consuming. Various advanced CNN architectures can detect and classify CT images. However, most require high computational costs and are not designed for commercial use. The study aims to automatically detect and classify ‘COVID-19 pneumonia’, 'normal', and ‘pneumonia’ lung CT images using transfer learning of the pre-existing CNN models and the proposed model. The study proposed a DL model inspired by ResNeXt and Inception to create a synergistic effect. The research conducted binary classification to compare the results of the existing models and multi-classification to compare the existing models and the proposed model. The existing models are DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, and ResNeXt 101. The dataset was collected from medRxiv, bioRxiv, NEJM, JAMA, Lancet, and the China National Centre of Bioinformatics. A comprehensive dataset was subdivided into training, validation, and testing. The images were then pre-trained using existing CNN architectures. Pre-trained models that had been fine-tuned extracted the features from the CT images. The research study applied transfer learning and deterministic concepts to the existing and proposed models to evaluate and compare their results. The proposed model is also designed to capture nuanced features indicative of COVID-19 infection, inspired by ResNeXt 101 and Inception, to achieve a lightweight and efficient CNN model. Data augmentation was applied to the dataset to introduce variations of unseen data to the proposed model. The study indicated the results for binary classification, ResNeXt 101, which obtained the best results among the existing CNN architectures. It acquired the highest sensitivity, specificity, precision, negative predicted value (NPV), accuracy, and F1-score of 0.9571, 1.0000, 1.0000, 0.9639, 0.9800, and 0.9781. The experimental results for multi-class classification showcase the efficacy of the proposed model, achieving an impressive accuracy, precision, recall, and F1-score of 0.9980. The proposed model has 7,724,523 parameters, 11 times less than ResNeXt 101 while having similar accuracy. The research depicted the possibilities of the proposed model in aiding medical diagnosis, especially in COVID-19 pneumonia detection using CT images. In conclusion, the proposed model and other developed models from the thesis offer promising tools for healthcare professionals to identify and detect COVID-19 pneumonia CT images early. It encourages the application of CNN architectures as a diagnostic aid for radiologists, particularly during the pandemic.

Keywords: convolutional neural network, deep learning, computed tomography, COVID-19, coronavirus

PENGESANAN PNEUMONIA COVID-19 PADA IMEJ TOMOGRAFI KOMPUTER MENGGUNAKAN MODEL PEMBELAJARAN DALAM RINGAN

ABSTRAK

SARS-CoV-2, juga dikenali sebagai COVID-19, ialah penyakit pernafasan berjangkit baru yang ditemui pada 2019 dan menyebabkan wabak di seluruh dunia yang meragut banyak nyawa. Wabak virus itu pada mulanya ditemui pada Disember 2019, Wuhan, China. Ia serta-merta meningkat menjadi krisis antarabangsa, menyebabkan penyakit berleluasa, kematian, dan gangguan sosio-ekonomi yang ketara. Pengenalpastian COVID-19 yang pantas dan tepat adalah penting dalam wabak global yang berterusan. Salah satu daripada pelbagai peranti perubatan ialah pengimejan tomografi berkomputer (CT), yang telah menunjukkan janji dalam aplikasinya untuk mengesan corak tersendiri yang berkaitan dengan kemerosotan tisu paru-paru. Amalan biasa adalah bergantung kepada ahli radiologi untuk mendiagnosis imej CT, yang memakan masa. Pelbagai seni bina CNN canggih boleh mengesan dan mengklasifikasikan imej CT. Walau bagaimanapun, kebanyakannya memerlukan kos pengiraan yang tinggi dan tidak direka untuk kegunaan komersial. Kajian ini bertujuan untuk mengesan dan mengklasifikasikan secara automatik imej CT paru-paru 'COVID-19 pneumonia', 'normal' dan 'pneumonia' menggunakan pembelajaran pemindahan model CNN yang sedia ada dan model yang dicadangkan. Kajian itu mencadangkan model DL yang diilhamkan oleh ResNeXt dan Inception untuk mencipta kesan sinergistik. Penyelidikan ini menjalankan klasifikasi binari untuk membandingkan keputusan model sedia ada dan pelbagai klasifikasi untuk membandingkan model sedia ada dan model yang dicadangkan. Model sedia ada ialah DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152 dan ResNeXt 101. Set data dikumpulkan daripada medRxiv, bioRxiv, NEJM, JAMA, Lancet dan Pusat Bioinformatik Kebangsaan China. Set data komprehensif telah dibahagikan kepada latihan, pengesahan dan ujian. Imej-imej itu kemudiannya dilatih terlebih dahulu menggunakan seni bina CNN sedia ada. Ciri-ciri imej CT telah diekstrak oleh model pra-latihan yang telah diperhalusi. Kajian penyelidikan menggunakan pembelajaran pemindahan dan konsep deterministik dalam model sedia ada dan yang dicadangkan untuk menilai dan membandingkan keputusannya. Model yang dicadangkan juga direka bentuk untuk menangkap ciri-ciri yang menunjukkan jangkitan COVID-19 yang diilhamkan oleh ResNeXt 101 dan Inception untuk mencapai model CNN yang ringan dan cekap. Pembesaran data telah digunakan pada set data untuk memperkenalkan variasi data yang tidak kelihatan kepada model yang dicadangkan. Kajian itu menunjukkan keputusan untuk klasifikasi binari, ResNeXt 101, yang memperoleh hasil terbaik di kalangan seni bina CNN sedia ada. Ia memperoleh kepekaan, kekhususan, ketepatan, nilai ramalan negatif (NPV), ketepatan dan skor F1 tertinggi sebanyak 0.9571, 1.0000, 1.0000, 0.9639, 0.9800 dan 0.9781. Keputusan eksperimen untuk klasifikasi berbilang kelas mempamerkan keberkesanan model yang dicadangkan, mencapai ketepatan, dan skor F1 yang mengagumkan sebanyak 0.9980. Model yang dicadangkan mempunyai 7,724,523 parameter, 11 kali kurang daripada ResNeXt 101 sambil mempunyai ketepatan yang sama. Penyelidikan itu menggambarkan kemungkinan model yang dicadangkan dalam membantu diagnosis perubatan, terutamanya dalam pengesanan pneumonia COVID-19 menggunakan imej CT. Kesimpulannya, model yang dicadangkan dan model lain yang dibangunkan daripada tesis menawarkan alat yang menjanjikan untuk profesional penjagaan kesihatan untuk mengenal pasti dan mengesan imej CT pneumonia COVID-19 lebih awal. Ia menggalakkan penggunaan seni bina CNN sebagai bantuan diagnostik untuk ahli radiologi, terutamanya semasa wabak.

Kata kunci: rangkain neural berlingkaran, pembelajaran dalam, imbasan tomografi berkomputer, COVID-19, koronavirus

ACKNOWLEDGEMENTS

I want to express my gratitude to the Faculty of Engineering, Universiti Malaya, for offering a Doctor of Philosophy and allowing me to fulfil the requirements of a Doctor of Philosophy. I learned, improved my skills, and was exposed to different software and practical skills, which were helpful guidance for my growth and development throughout my research experience.

First, I would like to express my profound appreciation to my supervisor, Assoc. Prof. Ir. Ts. Dr. Chuah Joon Huang, for allowing me the privilege of working under his guidance and for his perpetual patience and encouragement throughout my PhD journey. I want to convey my heartfelt gratitude to my supervisor Assoc. Prof. Ir. Ts. Dr. Lai Khin Wee for his unwavering support and valuable insights on the medical imaging perspective of my research. I am very thankful for his time and effort in guiding me to complete my project. My research would be incomplete without his advice, patience and contribution.

In addition, I want to extend my sincere gratitude to my superiors, colleagues, and friends at Universiti Malaya: Dr. Wee Hin Boo, Shazia Anis and Khaled Nedal, for their companionship, assistance and support throughout my PhD. I appreciate the time spent sharing insights and insightful criticism for my research to succeed.

Lastly, on a personal note, I want to express my deepest gratitude to my family members and loved ones (Lee Zhe Heng, Pius Lee, Peter Low, Rosi Chan, Nellie Low, Bonaventure Low, and Ignatius Low) for their unwavering support and encouragement as I pursued this PhD.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS AND ABBREVIATIONS

α	:	Alpha
β	:	Beta
δ	:	Delta
γ	:	Gamma
ACC	:	Accuracy
ACE2	:	Angiotensin-converting Enzyme 2
ADAM	:	Adam Optimiser Algorithm
AI	:	Artificial Intelligence
ALA	:	American Library Association
AMP	:	Automatic Mixed Precision
AP	:	Anterior-posterior
ARDS	:	Acute Respiratory Distress Syndrome
ART	:	Antigen Rapid Test
AT&T	:	American Telephone and Telegraph
BA.1	:	Omicron Variant
BA.4	:	Omicron Variant
BA.5	:	Omicron Variant
BBIBP	:	Beijing Institute of Biological Products
CAD	:	Computer-Aided Diagnosis
CDC	:	Centres for Disease Control and Prevention
CD13	:	Cluster of Differentiation 13
CHO	:	Chinese Hamster Ovary
CI	:	Confidence Interval
CIGB	:	Centre for Genetic Engineering and Biotechnology (Cuba)

CLA	:	Cross-level Attention
CNCB	:	China National Centre for Bioinformation
CNN	:	Convolution Neural Network
COVID-19	:	Coronavirus Disease 2019
CoVs	:	Coronaviruses
CPU	:	Central Processing Unit
CrossViT	:	Cross-attention Vision Transformer
CSA	:	Cross-scale Attention
CT	:	Computer Tomography
CXR	:	Chest Radiography
DCNN	:	Deep Convolutional Neural Network
DL	:	Deep Learning
DNA	:	Deoxyribonucleic Acid
DPP4	:	Dipeptidyl Peptidase IV
FAKHRAVC	:	COVID-19 Vaccine Developed in Iran
F1	:	Sørensen–Dice coefficient
FC	:	Fully Convolutional
FN	:	False Negative
FP	:	False Positive
GB	:	Gigabyte
GGO	:	Ground Glass Opacities
GHz	:	Giga Hertz
GISAID	:	Global Initiative on Sharing All Influenza Data
GPU	:	Graphics Processing Unit
GSK	:	UK Biopharma Company
HU	:	Hounsfield Unit

ICTV	:	International Committee on Taxonomy of Viruses
KNN	:	K-Nearest Neighbour
Mbconv	:	Building Block of Efficient Nets, an Inverted Residual Block
MERS	:	Middle East Respiratory Syndrome
ML	:	Machine Learning
MLP	:	Multilayer Perceptron
NAATs	:	Nucleic Acid Amplification Tests
NASNet	:	Neural Search Architecture (NAS) Network
NPV	:	Negative Predictive Value
NVX	:	Nuvaxovid
PA	:	Posterior-Anterior
PIL	:	Python's Pillow
PPV	:	Positive Predictive Value
RAM	:	Random Access Memory
ResBlock	:	Residual Block
ReLU	:	Rectified Linear Unit
ResNet	:	Residual Network
ResNeSt	:	Residual Neural Network with Split-Attention
RF	:	Random Forest
RGB	:	Red, Green and Blue
RIBSP	:	Research Institute for Biological Safety Problems
RNA	:	Ribonucleic Acid
RT-PCR	:	Reverse Transcript Polymerase Chain Reaction
SA	:	Sialic Acid
SARS	:	Severe Acute Respiratory Syndrome
SE	:	Squeeze-and-Excitation

SENet	:	Squeeze-and-Excitation Network
SGD	:	Stochastic Gradient Descent
SK	:	South Korea Vaccine Company
SVM	:	Support Vector Machine
TFLOPS	:	One Trillion Floating-point Operations per Second
TN	:	True Negative
TNR	:	True Negative Rate
TP	:	True Positive
TPR	:	True Positive Rate
UK	:	United Kingdom
USA	:	United States of America
VGG	:	Visual Geometry Group
ViT	:	Vision Transformer
VLP	:	Virus-like Particles
WHO	:	World Health Organisation
XAI	:	Explainable Artificial Intelligence
XOR	:	Exclusive OR
ZFNet		Zeiler and Fergus Net

LIST OF APPENDICES

CHAPTER 1: INTRODUCTION

1.1 Background

COVID-19 is an infectious disease, the fifth flu pandemic after the 1918 pandemic. The name of the virus is SARS-CoV-2. COVID-19 is one of seven coronaviruses discovered by humankind that cause illness in humans and animals. Besides COVID-19, SARS and MERS are other lethal coronaviruses that cause minor pandemics in certain third-world countries. Other coronaviruses create mild illness called rhinoviruses (Kandola, 2020). Coronaviruses are under the subfamily *Orthocoronavirinae*, in*onaviridae*, order *Nidovirales* and realm *Riboviria* (Executive Committee of ICTV, 2019) (Fan et al., 2019). The initial report indicated that while the COVID-19 viral pneumonia differed from the actual SARS coronaviruses, it shared 88% sequence identity with two bat-derived SARS-like viruses (Lu et al., 2020).

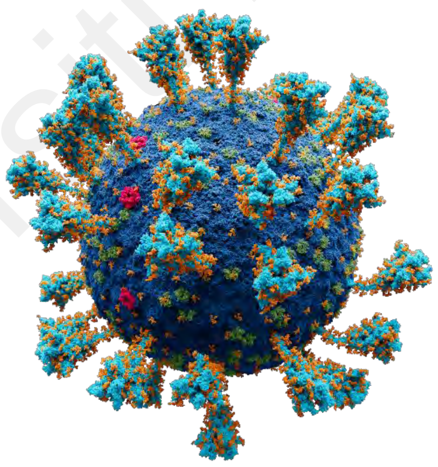


Figure 1.1: Scientifically Accurate Coronavirus Model (SARS-CoV-2) (Solodovnikov & Arkhipova, 2021).

One of the main parts of coronaviruses is a nucleocapsid with helical symmetry and a positive-sense single-stranded RNA genome, as described in Figure 1.1 (Chen et al., 2007). Its genome size ranges from roughly 26 to 32 kilobases, and it is one of the most significant RNA viruses. Each coronavirus particle consists of 74 surface spikes, approximately 20 nm long. On the surface of the lipid bilayer envelope, attach the glycoprotein spikes. The spike protein interacts with the host cell receptor binding of an infected animal or human and undergoes membrane fusion. It is part of the order Nidovirales, family Coronavirdiae, and the subfamily Coronavirinae. The COVID-19 pneumonia RNA genome comprises a 5' methyl-guanosine cap, poly (A)-tail, and 29,903 nucleotides. It indicated that bats and rodents are the genetic foundations of α and β coronaviruses, and avian species are the genetic bases of most δ and γ coronaviruses (Yen-Chin Liu, 2020).

Coronaviruses (CoVs) are viruses responsible for various mild-to-severe human illnesses. It was named for the crown-like spikes on its surfaces. Initially, six coronaviruses caused disease in humans. The set of coronaviruses included the alpha (α), beta (β), gamma (γ), and delta (δ) subgroups. α-coronaviruses (229E, NL63,) and β-coronaviruses (OC43, HKU1) were the most prevalent in humans (UK Research and Innovation, 2022). Table 1.1 describes the seven types of coronaviruses that are harmful to humanity, and COVID-19 was the latest to be discovered (UK Research and Innovation, 2020). In December 2019, in Wuhan City, Hubei Province, China, an unknown aetiology of a lower respiratory tract febrile illness was initially identified (Moore, 2021). This pulmonary disease has been declared Coronavirus Disease 2019 (COVID-19) by the World Health Organisation (WHO). It is the latest coronavirus to be found, infecting people worldwide.

Table 1.1: Seven Types of Coronaviruses (UK Research and Innovation, 2020).

Type of Coronavirus	Type of Illness
SARS-CoV-2	COVID-19
SARS-CoV	SARS
MERS-CoV	MERS
HCoV-NL63	Common Cold
HCoV-229E	
HCoV-OC43	
HKU1	

All coronaviruses originate in animals and are later transmitted to humans. These animals are the coronavirus's natural hosts. Bats are unquestionably significant and the source of α and β-coronaviruses. They will likely be the first hosts of the common cold, SARS, and MERS viral pneumonia (Yen-Chin Liu, 2020). Furthermore, rodents were likely the source of HCoV-OC43 and HKU1 (Yen-Chin Liu, 2020). As intermediary hosts, domestic animals can contract diseases that spread viruses from wild animals to people. For instance, the SARS-CoV and MERS-CoV viruses have infected camels and masked palm civets, respectively (Yen-Chin Liu, 2020). By comparing the genome's full-length SARS-CoV-2, RNA sequences at the beginning of the COVID-19 outbreak only had 79.6% RNA sequence identity with SARS-CoV (Yen-Chin Liu, 2020). However, it practically has a 96.2% similarity at the whole-genome level to Bat-CoV RaTG13, which was initially identified in *Rhinolophus affinis* from Yunnan Province, more than 1500 kilometres from Wuhan (Yen-Chin Liu, 2020). Although bats are the most probable SARS-CoV-2 reservoir hosts, it is still unclear if Bat-CoV RaTG13 jumps straight to people or passes through intermediary hosts to enable animal-to-human transmission (Yen-Chin Liu, 2020). Researchers did not obtain an intermediate host sample in

Wuhan's initial cluster of COVID-19 infections. Table 1.2 illustrates the classification of human coronaviruses.

Table 1.2: Classification of Human Coronaviruses (Yen-Chin Liu, 2020).

Strain	Year Found	Cellular Receptor	Natural Host	Intermediate Host	Symptoms
HcoV-229E	1966	Aminopeptidase N (CD13)	Bat	Camelids	Minor
HcoV-OC43	1967	9-O-Acetylated sialic acid (SA)	Rodent	Cattle	Minor
SARS	2003	ACE2	Bat	Masked palm civets	Major
HcoV-NL63	2004	ACE2	Bat	Unknown	Minor
HcoV-HKU1	2005	9-O-Acetylated sialic acid (SA)	Rodent	Unknown	Minor
MERS	2012	DPP4	Bat	Dromedary camels	Major
COVID-19	2019	ACE2	Bat	Pangolin	Major

COVID-19 is similar to viral pneumonia lung inflammation in the alveoli (WebMD, 2020). The inflammation brought on by COVID-19 pneumonia destroys the cells and tissues of the air sac line in the lungs, causing the respiratory system to fail (WebMD, 2020). The inflammation gradually builds up fluid and pus in the alveoli, causing breathing difficulties. The patient exhibits chest pains, stuffy nose, weariness, shortness of breath, cough, fever, fatigue, and other symptoms. The symptoms are comparable to viral pneumonia and include acute respiratory distress syndrome (ARDS) (Seladi-Schulman, 2020), rapid heart rate, dizziness, and heavy perspiration (WebMD, 2020). Henceforth, it is challenging to accurately detect a patient with COVID-19 pneumonia or other variations of viral pneumonia via medical images. Deep learning models can aid medical experts in executing time-consuming and labour-intensive work, such as determining CT images for symptoms of COVID-19 pneumonia and identifying the severity of the infection (Chouhan et al., 2019). CNN architectures imitate human-level accuracy and precision in examining and segmenting medical images without human error. The intensive growth of CNN and AI has been actively applied to medical imaging. Its approaches have proven to achieve quality prediction capabilities mimicking the results of radiologists (Müller et al., 2020). However, CNN models cannot replace medical professionals like physicians, radiologists, and clinicians in diagnosing medical images.

The COVID-19 virus alleviated environmental concerns worldwide as the government announced a lockdown, and guidelines for social distancing, quarantine, and washing hands are frequently advised to the general public nationwide. With the travel restrictions implemented worldwide, the virus has been transmitted exponentially, with confirmed cases transmitted in Europe, the United Kingdom, the United States, and worldwide (World Health Organization, 2020). The coronavirus possessed a different nucleic acid sequence that was unbeknownst to humankind. The virus was detected in lung fluid, blood, and throat swab samples. The medical and scientific communities found themselves racing against time to stop the disease from spreading and develop vaccines immediately. Medical imaging has been identified as an essential tool to help the scientific community provide early disease recognition and isolate affected patients. It is a diagnostic tool for monitoring disease progression and defining characteristics of the coronavirus's acute and chronic phases (Al-Tawfiq et al., 2014).

In the past, similar pulmonary syndromes have been identified as the cause of other coronavirus families. Severe acute respiratory syndrome (SARS) and Middle East respiratory syndrome (MERS) are the most famous coronaviruses that primarily and severely affect the population worldwide. The last known SARS outbreak was reported in 2003, and it had been contained. The imaging features of SARS and MERS are inconsistent and non-specific. The imaging results of COVID-19 pneumonia were also deemed to be ambiguous. Researchers are trying to further categorise the imaging features of COVID-19 images. As of November 9, 2022, COVID-19 cases were confirmed in 230 countries and territories worldwide, with 638,435,847 cases and total deaths of 6,608,101, a mortality rate of 1.55% (American Library Association (ALA), 2022). The United States of America (USA), India, Brazil, France, the United Kingdom (UK), Russia, Turkey, Italy, and Spain are the top 10 nations where COVID-19 pneumonia cases have been documented.

Automated detection is necessary for early detection before symptoms manifest, as asymptomatic transmission might become the source of infection. The results of CT scan diagnostic abilities can be supported by AI-based detection technologies, which further alleviate the workload on local medical systems for both the clinical and public health sectors. The COVID-19 vaccine has been developed to provide immunity against ARDS. Before COVID-19, knowledge about coronaviruses' structures, like SARS and MERS, was well established. That established knowledge accelerated the progress of numerous COVID-19 inoculations during early 2020 and finally developed vaccines and distributed them at the end of 2020 (Li et al., 2020). The genetic sequence of SARS-CoV-2 was available through GISAID, and on March 19, 2020, the global pharmaceutical industry proclaimed its obligation to address and develop a vaccine for COVID-19 (Padilla, 2021). Since February 7, 2022, 10.22 billion COVID-19 vaccines have been distributed globally, and 23.37 million are administered daily (Ritchie et al., 2020). While various vaccines have been administered globally, prompt diagnosis and identification of medical images are essential. Hence, we should use advanced technology to eliminate this global epidemic.

1.1.1 Classification Pulmonary Computed Tomography Images

Over the years, various classifications of CT images in different body positions have been used in deep learning (DL) (Sugimori, 2018). CNN architectures were built and compared with AlexNet and GoogLeNet to sort CT scans of the brain, chest, abdomen, pelvis, and neck, using both plain and contrast-enhanced images (Sugimori, 2018). CNN architectures have successfully classified general document images (LeCun et al., 1998). The intricacies of the human anatomy are an obstacle for DL to conduct accurate classification (Roth et al., 2015). The classification of pulmonary CT images is significantly complex; only trained radiologists are certified to diagnose the images accurately. Pulmonary diseases often applied to CT scans for diagnosis are pulmonary embolism, lung cancer, pleural effusion, thoracic trauma, smoking and cancer risk cases, Chronic Obstructive Pulmonary Disease (COPD) (Cagle Jr et al., 2023), bronchiectasis, emphysema, and respiratory diseases. The application of CNN can perform several classifications on pulmonary CT images, like segmentation, detection, diagnosis, or prognosis.

1.1.2 Clinical Findings, Diagnostics, Testing and Vaccinations

During the pandemic, there are numerous simultaneous clinical implications for global healthcare professionals to prevent the spread of SARS-CoV-2. Understanding the evolutionary network and pathophysiology of the disease is crucial to determining how the new virus has adapted to humans and various contexts. Healthcare professionals need to find and identify the virus in the interim, then provide meticulous patient care and prevent further infection (Weissleder, Lee, Ko, & Pittet, 2020). The virus was initially revealed at the end of December 2019 from Wuhan, where the submitted sequence differed from the discovery at the beginning of April 2020 from North America, according to information provided by COVID-19 viral sequences uploaded into the GISAID record in January 2020. Information on creating vaccines for various populations will come from analysing the geographical patterns of several COVID-19 variants. In contradiction to SARS and MERS, which have a greater death rate but stronger reinfection than community transmissibility, the COVID-19 virus can spread rapidly in the population (Yen-Chin Liu, 2020).

The COVID-19 virus incubation period differs from one another but approximately ranges from 1 to 14 days. The average incubation period for COVID-19 was shown to be 5.2 days, with a 95% confidence interval (CI) (Yen-Chin Liu, 2020; Adaloglou, 2021). A person infected with COVID-19 is highly contagious during that period (Li et al., 2020). Pneumonia affected every patient, and almost half of them also had dyspnea. After admission, COVID-19 pneumonia patients exhibited arrhythmia, severe cardiac damage, decreased renal function, and abnormal liver function (Yen-Chin Liu, 2020). Research in Wuhan, China, indicated that 36.4% of the 214 COVID-19 pneumonia-infected patients suffered from neurologic symptoms.

It was further discovered that a neurovirulent COVID-19 viral infection of the olfactive system was the cause of the impaired capacity to taste or smell observed in some patients (Yen-Chin Liu, 2020). A wide range of symptoms can be found in an infected patient, together with fever, cough, chills, difficulty breathing, lack of breath, chest ache, sore throat, vomiting, nausea, and diarrhoea (Coronavirus disease (COVID-19), 2020). Infected patients might have average white blood cell counts (Guan et al., 2020; Wang et al., 2020). Some patients experience indications of hemoptysis (Guan et al., 2020; Wang et al., 2020) and can be symptomless, while the rapid development of the disease may cause organ dysfunction. The following examples of organ dysfunction may cause the body to experience extreme reactions: shock, acute respiratory disease syndrome (ARDS), acute cardiac and kidney injury, and even death. Henceforward, quarantine and isolation protocols were established based on the virus's incubation period in the body. Unvaccinated individuals or those who were not up-to-date in receiving their primary series of COVID-19 vaccines, followed by booster doses, have to quarantine, refrain from travelling, and be immediately tested.

As a result of the COVID-19 disease's rapid dissemination, diagnostic tools were created very quickly, enabling the early identification of this novel virus. A rapid test must be conducted once the patient is infected. The suggested specimen for molecular analysis is a nasopharyngeal swab. The CDC has approved oropharyngeal, mid-turbinate, and nasal swabs as acceptable specimen types of nasopharyngeal swabs (Yen-Chin Liu, 2020). Initially, the standard tool to diagnose COVID-19 was to detect the existence of COVID-19 viral pneumonia in a collected specimen using reverse transcript polymerase chain reaction (RT-PCR). The specimen is collected via nasal or throat swabs to detect the existence of COVID-19 viral pneumonia RNA (Ferrari & Angelis, 2020). Patients suspected of having a COVID-19 pneumonia infection have samples taken from their upper and lower respiratory tracts. The latest viral assessments are the nucleic acid amplification tests (NAATs) and antigen tests to distinguish the current infection with COVID-19.

NAATs are sensitive, specific, and accurate in diagnosing SARS-CoV-2 infection. It discovers one or more viral RNA genes to indicate the status of the infection. NAATs are commonly conducted in the laboratory and may take up to 3 days to receive the results. There are also point-of-care test options for some NAATs, with results available in 15 to 45 minutes. The rapid version of the NAATs is not a hypothetical result and can confirm whether an individual is infected (Centers for Disease Control and Prevention, 2021). The recommended samples to collect and test are samples from the upper respiratory system. Antigen and antibody tests are other test methods that can be performed rapidly. Antigen tests detect a specific viral antigen but are less sensitive than most NAAT rapid tests. It is usually applied as a screening tool to detect who is contagious immediately. Antibody tests, also known as serology tests, are not recommended to diagnose current infections. It should not determine whether a person is immune to reinfection. Oral specimens are inappropriate for confirmatory testing (Centers for Disease Control and Prevention, 2021).

Differential diagnosis reports from several regions with high COVID-19 infections depicted chest CT as one of the diagnostic tools (Park et al., 2021). The application of chest CT is relevant, especially in a hospital where inpatients are high, and the suspicion of the patient with COVID-19 is undetectable by the RT-PCR swab test (Park et al., 2021). Moreover, patients with respiratory symptoms might have underlying respiratory diseases and need to perform a chest CT to diagnose the respiratory issue. Lately, meta-analysis has indicated that pulmonary thrombus and embolism are complications of COVID-19. Diagnostic analysis indicates typical CT abnormalities, such as bilateral pulmonary parenchymal ground glass and consolidated pulmonary opacities that were occasionally spherical and distributed throughout the peripheral lungs (Yen-Chin Liu, 2020). These patients are admitted to the Intensive Care Unit (ICU) and have a CT pulmonary angiogram to prove the diagnosis.

Since 2020, vaccines have been developed, and several are currently in phases I/II. There are three principal methods for developing a vaccine: (1) the whole microbe, (2) the subunit, and (3) the genetic approach (Le et al., 2020). Figure 1.2 illustrates the vaccine platform divided into categories and subcategories.

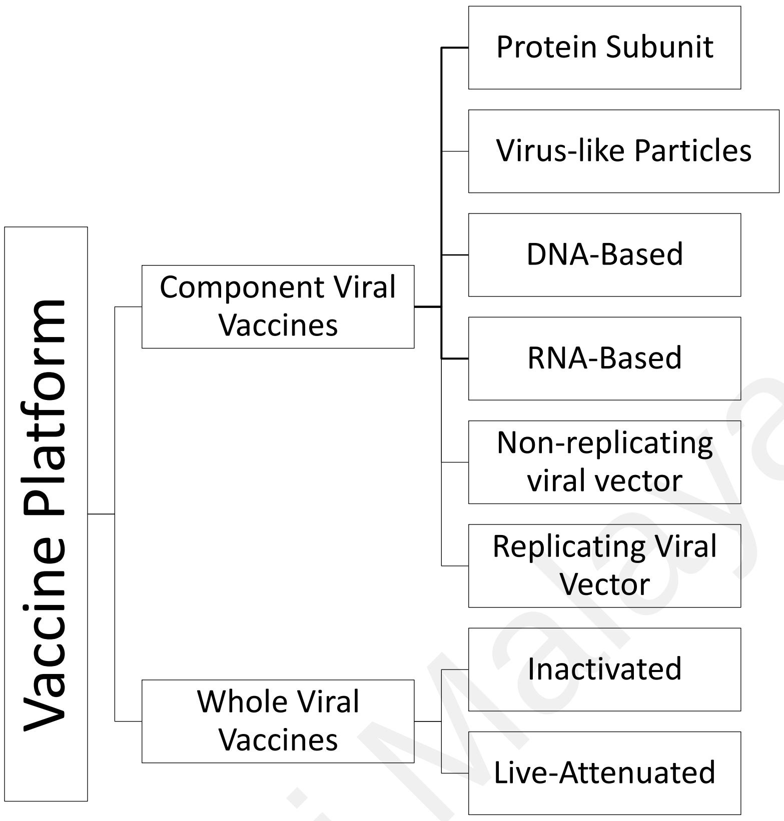


Figure 1.2: Types of Vaccines (VIPER Group COVID-19 Vaccine Tracker Team, 2022)

The most effective method to train the human immune system to recognise viruses or antigens is via vaccinations. The COVID-19 virus comprises spike protein on the surface as an antigen (VIPER Group COVID-19 Vaccine Tracker Team, 2022). When a person who has received the vaccine is exposed to the COVID-19 virus, their immune system will distinguish the viral antigen and mobilise to protect their health (VIPER Group COVID-19 Vaccine Tracker Team, 2022). As shown in Figure 1.2, COVID-19 vaccinations commonly fall into one of two categories: i) whole virus vaccines; ii) component viral vaccines. The component viral vaccines consist of five types: i) protein subunit, ii) virus-like particles (VLP), iii) DNA-based and RNA-based, iv) non-replicated viral vector, and v) replicating viral vector. At the same time, the whole virus vaccines is divided into inactivated and live-attenuated vaccines.

Protein subunit vaccines are the type that include a tiny portion of the isolated or purified viral protein, enabling the immune system to identify that virus fragment (National Library of Medicine, 2023). Viral genetic material, such as mRNA, is present in DNA- or RNA-based vaccinations, and it gives the body instructions on how to produce tiny virus pieces that the immune system can recognise (Shukla, 2021). The non-replicating viral vector vaccines transmit the genetic code for making tiny fragments of viral protein encapsulated inside a different harmless virus that cannot replicate itself (Guo et al., 2023). The immune system then recognises that viral protein fragment (VIPER Group COVID-19 Vaccine Tracker Team, 2022). Table 1.3 shows the types of vaccines differentiated by the development methodologies and administered in several respective countries worldwide.

Table 1.3: Vaccines Information (VIPER Group COVID-19 Vaccine Tracker Team, 2022).

Types of Vaccines	**Development Methodologies**	**Status**
Novavax: Nuvaxovid NVX-CoV2373	Protein Subunit	40 Countries
Serum Institute of India: COVOVAX (Novavax formulation)	Protein Subunit	6 Countries
Moderna: Spikevax mRNA-1273	RNA	88 Countries
Pfizer/BioNTech: Comirnaty Tozinameran, BNT162b2	RNA	149 Countries
Janssen (Johnson & Johnson) Ad26.COV2.S	Non-Replicating Viral Vector	113 Countries
Oxford/AstraZeneca: Vaxzevria AZD1222, ChAdOx1 nCoV-19	Non- Replicating Viral Vector	149 Countries
Serum Institute of India Covishield (Oxford/ AstraZeneca formulation)	Non-Replicating Viral Vector	49 Countries
Bharat Biotech: Covaxin BBV152	Inactivated	14 Countries
Sinopharm (Beijing) Covilo BBIBP-CorV (Vero Cells)	Inactivated	93 Countries
Sinovac CoronaVac	Inactivated	56 Countries
Anhui Zhifei Longcom Zifivax	Protein Subunit	4 Countries
Bagheiat-allah University of Medical Science Noora Vaccine	Protein Subunit	1 Country
Bharat Biotech iNCOVACC	Non-replicating Viral Vector	1 Country
Biological E Limited Corbevax	Protein Subunit	2 Countries
CanSino Convidecia	Non-Replicating Viral Vector	10 Countries
CanSino Convidecia Air	Non-Replicating Viral Vector	2 Countries
Center for Genetic Engineering and Biotechnology (CIGB) Abdala	Protein Subunit	6 Countries
Chumakov Center KoviVac	Inactivated	3 Countries
Gamaleya Gam-COVID-Vac	Non-Replicating Viral Vector	1 Country

Table 1.3 Continued.

Types of Vaccines	Development Methodologies	Status
Gamaleya Sputnik Light	Non-Replicating Viral Vector	26 Countries
Gamaleya Sputnik V	Non-Replicating Viral Vector	74 Countries
Gennova Biopharmaceuticals Limited GEMCOVAC-19	RNA	1 Country
Health Institutes of Turkey Turkovac	Inactivated	1 Country
Instituto Finlay de Vacunas Cuba Soberana 02	Protein Subunit	4 Countries
Instituto Finlay de Vacunas Cuba Soberana Plus	Protein Subunit	2 Countries
Livzon Mabpharm Inc V-01	Protein Subunit	1 Country
Medicago Covifenz	VLP	1 Country
Medigen MVC-COV1901	Protein Subunit	4 Countries
Moderna Spikevax Bivalent Original/Omnicron BA.1	RNA	38 Countries
Moderna Spikevax Bivalent Original/Omnicron BA.4/BA.5	RNA	33 Countries
National Vaccine and Serum Institute Recombinant SARS-COV-2 Vaccine (CHO Cell)	Protein Subunit	1 Country
Organisation of Defensive Innovation and Research FAKHRAVC (MIVAC)	Inactivated	1 Country
Pfizer/BioNTech Comirnaty Bivalent Original/Omicron BA.1	RNA	35 Countries
Pfizer/BioNTech Comirnaty Bivalent Original/Omicron BA.4/BA.5	RNA	33 Countries
PT Bio Farma IndoVAC	Protein Subunit	1 Country
Razi Vaccine and Serum Research Institute Razi Cov Pars	Protein Subunit	1 Country
Research Institute for Biological Safety Problems (RIBSP) QazVac	Inactivated	2 Countries
Sanofi/GSK VidPrevtyn Beta	Protein Subunit	30 Countries
Shenzhen Kangtai Biological Products Co KCONVAC	Inactivated	2 Countries

Table 1.3 Continued.

Types of Vaccines	Development Methodologies	Status
Shifa Pharmed Industrial CoCOVIran Barekat	Inactivated	1 Country
Sinopharm (Wuhan) Inactivated (Vero Cells)	Inactivated	2 Countries
SK Bioscience Co Ltd SKYCovione	Protein Subunit	1 Country
Takeda TAK-019 (Novavax formulation)	Protein Subunit	1 Country
Takeda TAK-919 (Moderna formulation)	RNA	1 Country
Valneva VLA2001	Inactivated	33 Countries
Vaxine/CinnaGen Co. SpikoGen	Protein Subunit	1 Country
Vector State Research Center of Virology and Biotechnology Aurora-CoV	Protein Subunit	1 Country
Vector State Research Center of Virology and Biotechnology EpiVacCorona	Protein Subunit	4 countries
Walvax AWcorna	RNA	1 Country
Zydus Cadila ZyCoV-D	DNA	1 Country

At least one nation has approved the above vaccinations. Vaccines that have received approval have been authorised, licenced, designated for emergency use, or made accessible outside clinical trials (VIPER Group COVID-19 Vaccine Tracker Team, 2022).

1.2 Problem Statement

At the beginning of the COVID-19 diagnosis, RT-PCR tests are the critical diagnostic methods to identify COVID-19 in a patient. Early diagnosis and isolation of infected COVID-19 patients are essential for preventing the severe spread of the infection and supporting triage, clinical, and public health to evaluate and monitor potential patients that may decrease operational strain on the healthcare sector to cope with the overwhelming COVID-19 pandemic (Rohanian et al., 2023). Molecular assays like RT-PCR are currently used to support effective clinical and public use for detecting various respiratory infections, resulting in institutions' viral outbreaks (CDC, 2019). The physical appearance of the test kit is similar to a pregnancy test; instead of using urine to determine if an individual is COVID-19 positive, the RT-PCR uses oral fluids like saliva, phlegm, or nasal mucus, but its low sensitivity and false negative rate require alternative methods to diagnose COVID-19 (CDC, 2019).

Computed tomography (CT) could also be one of the alternatives to detect if the patient is diagnosed with COVID-19 pneumonia. With the increase in COVID-19 pneumonia cases, CT scans can be implemented as the primary screening and diagnostic tool for COVID-19 pneumonia detection, as they quick and easy to perform, but the job process is challenging for physicians. Even though the resolutions of the CT images increase, producing high COVID-19 pneumonia detection capabilities, the evaluation, annotation, and segmentation of the pulmonary lungs on the medical images are labour-intensive, monotonous, and time-consuming procedures conducted by radiologists (Ronneberger et al., 2015). The process of annotation, segmentation, and classification by a person is a burden (Ronneberger et al., 2015). A patient requires numerous CT images to support the claims and accurately detect COVID-19 pneumonia. The problem of procuring accurate assessments arose when medical professionals were required to deal with a colossal number of patients in a short period. Clinical experience is also a significant factor influencing the quality of annotating medical images (Cozzi et al., 2020). However, CT images interpreted by professionals may also develop issues to a certain percentage, such as human error, practical predisposition, variations among medical professionals, and interpretation discrepancy (Lee et al., 2022).

Implementing AI-based algorithms to automate disease recognition is vital to accurately diagnosing pulmonary CT scans. The technology can assist clinicians in triage valuation, clinical investigation, and diagnostic detection (Ronneberger et al., 2015). However, most deep-learning models that classify CT images have many parameters that require high computational costs and time to run and produce results (Adaloglou, 2021). Moreover, the higher number of parameters in the CNN model does not imply it is more accurate in classification (Adaloglou, 2021). Therefore, a lightweight DL model is required to classify COVID-19 CT images accurately.

1.3 Aim & Objectives of the Study

This research aims to study, design, and develop a deep learning proposed model that can classify 'COVID-19 pneumonia', 'normal', and other 'viral pneumonia' pulmonary CT images.

The objectives of this research are:

1. To evaluate deep learning models for classifying 'COVID-19 pneumonia', 'normal', and other 'viral pneumonia' pulmonary CT images.
2. To design a lightweight deep learning model comparable with the preexisting models' capabilities.
3. To evaluate the performances of established transfer learning models and compare them with the design proposed model with a confusion matrix and performance metrics.

1.4 Scope of the Study

This research aims to design and develop a proposed CNN architecture to automate binary and multi-classification 'COVID-19 pneumonia', 'normal', and 'other viral pneumonia' pulmonary CT images. A thorough investigation of COVID-19, bacterial, and other viral pneumonia clinical and radiological information is performed. The study is categorised into three stages. The initial stage includes a feasibility study of the CT images using CNN algorithms and classifying them. Various CNN architectures are studied, and the related research is included and summarised. The second stage is the development of CNN classification models, starting by pre-processing the CT images. Data augmentation is applied to upsurge the quantity and introduce variations of unseen datasets for training and validation.

The final stage is to design a proposed CNN model and compare the proposed model with various established CNN architectures via transfer learning. Modifications to the hyperparameters in the CNN models are performed to ensure the models are vigorous and consistent.

1.5 The Organisation of the Thesis

The thesis is separated into five chapters. Chapter 1 presents the background research, problem statements, aim and objectives, and scope of work. The research background provides insight into COVID-19 pneumonia and its development as a worldwide epidemic. The problem statement highlights the trials in detecting COVID-19 pneumonia using CT imaging. Also, the scope describes the research's general roadmap.

Chapter 2 is the literature review, a compilation of various research studies comprising clinical studies about COVID-19 and the current diagnostic tests and vaccines. The literature review also includes the study of AI in medical imaging, explaining various CNN architectures implemented on pulmonary CT images to classify 'COVID-19 pneumonia', 'normal', and 'other viral pneumonia' images.

Chapter 3 is the research methodology that depicts the data acquisition, pre-processing, data augmentation, transfer learning, proposed workflow, model training, validation, testing, and model designing. Chapter 4 contains the CNN models' training, validation, and test results, which are discussed accordingly. Finally, Chapter 5 concludes the research based on the findings, including limitations and recommendations for future improvement.

CHAPTER 2: LITERATURE REVIEW

2.1 Coronavirus Disease 2019 (COVID-19)

2.1.1 Radiological Perspective of COVID-19

Computer tomography (CT) images have been ideal for diagnosing COVID-19 (Apostolopoulos & Mpesiana, 2020). There are numerous diagnostic challenges in applying CT scans to detect the early stages of COVID-19 pneumonia symptoms. Radiography imaging plays a vital role in revealing the COVID-19 pneumonia indications as ground-glass opacities with peripheral, bilateral, and primary basal distributions (Rubin et al., 2020). High-resolution CT scans can detect small lung lesions, visualise the inflammation, and signal the radiologist regarding the patient's condition. Besides, CT images can track and monitor the progress of a treated patient. SARS and MERS have similar virulent aspects, clinical indications, and imaging structures, which are significant advantages that may assist in identifying COVID-19 imaging in the critical and chronic stages of the patients (Hosseiny et al., 2020). However, the WHO rejected CT findings without RT-PCR tests to confirm the validity of the images (Zu et al., 2020). COVID-19 has the primary symptoms of pneumonia. The chest X-ray images (CXR) for COVID-19 describe patchy or diffuse asymmetric air space opacity characteristics (Chen et al., 2020). Multifocal ground-glass opacities and consolidations were seen on CT scans, along with peripheral lung predilection and patchy ground-glass opacities on both sides (Chung et al., 2019). The following are the characteristics of COVID-19 patients' CT images: a) ground-glass opacities; b) consolidations; c) the number of lobes affected by ground-glass opacities or consolidations; d) total severity scores; e) the presence of nodules; f) pleural effusion; g) thoracic lymphadenopathy; h) underlying lung diseases; i) other abnormalities in the lungs (Zu et al., 2020). Ground-glass-like lung appearances are most likely caused by the severe lung inflammation that prevents lung cells from exchanging oxygen and carbon dioxide during the SARS-CoV-2 infection (Yen-Chin Liu, 2020). Based on a study by (Park et al., 2021), chest CT has a sensitivity of 90% to diagnose COVID-19. However, the specificity varies and falls from 25% to 83%. The application of chest CT is limited even in hospitalised patients, where it is only performed with rapid tests or blood tests with high false-positive rates (Park et al., 2021).

Computed tomography (CT) imaging is essential for the ancillary analysis at the preliminary screening. The finding is subsequently verified by the positive outcomes of the NAATs performed on blood samples or respiratory tract specimens using RT-PCR. When the viral load is low, the detection rate is poor, which indicates false-negative results, severely restricting this diagnosis method. The positive diagnosis for COVID-19 viral pneumonia can be assessed and results determined, and while pulmonary CT images may demonstrate the disease progression, neither the severity of the pulmonary condition nor the disease progression can be confirmed. As a result, the patient needs to be separated and treated immediately if they have COVID-19 suspicion and have positive imaging results but negative NAAT results (Dai et al., 2020).

2.1.2 COVID-19 vs. Viral Pneumonia

Computed tomography images (CT) are used as an essential complementary method to diagnose COVID-19 pneumonia when other resources like RT-PCR or sequencing are scarce in the emergency setting (Hani et al., 2020). Generally, the characteristics of COVID-19 chest CT images are the existence of ground glass opacities (GGO) usually found at the peripheral and sub-pleural distribution (Salehi et al., 2020; Cheng et al., 2020). The majority of COVID-19 patients were found to have GGO in their lower lobes, which was combined with focal consolidations and overlaid intralobular reticulations to form an odd pavement pattern.

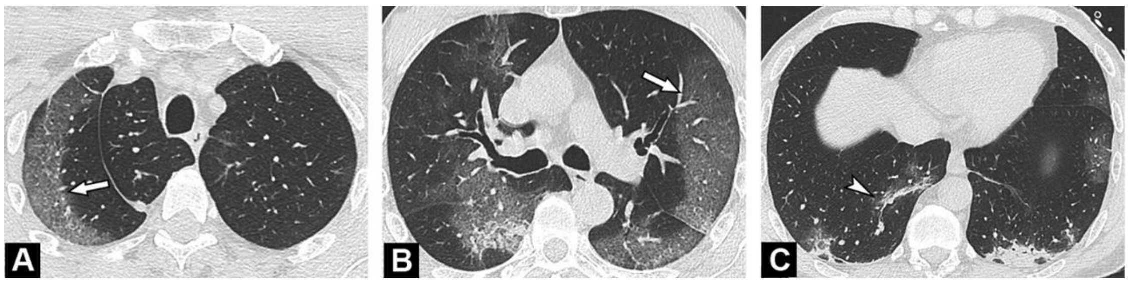


Figure 2.1: COVID-19 Chest CT Findings - A) Peripheral GGO, B) Peripheral GGO Upper Portion of the Lungs, C) Lower Lobes Linear Consolidations (Hani et al., 2020)

Figure 2.1 describes an unenhanced COVID-19 CT image of a 55-year-old male. The patient was tested twice using RT-PCR, and the results were negative. It was only after reviewing of the CT findings that a third test was conducted and finally, the result showed positivity (Hani et al., 2020). Multifocal bilateral patchy ground-glass opacities (GGOs), consolidation with the interlobular septum, and vascular thickening, which is typically found on the periphery of the lungs, are the most frequently found characteristics on chest CT scans (Li M. , 2020). GGOs and consolidation have occurrence rates of roughly 86% and 29%, respectively, and their prevalent form is patchy, round opacities, followed by triangular and linear ones (Kanne, 2020). Many viruses, including the respiratory syncytial virus (RSV), influenza virus, parainfluenza virus, adenovirus, measles virus, SARS, MERS, and COVID-19 virus, are frequently responsible for viral pneumonia (Hadjiliadis et al., 2022). The lungs' morphology, distributions, number of lobes, particular focal signs, and extrapulmonary symptoms are assessed by the chest CT features (Niu et al., 2021).

Unless they have a persistent illness, older adults and younger children are more susceptible to viral pneumonia because their immune systems are not as developed as those of people in their prime (Freeman & Leigh, 2023). Many viruses, including the respiratory syncytial virus (RSV), influenza virus, parainfluenza virus, adenovirus, measles virus, SARS, MERS, and COVID-19 virus, are frequently responsible for viral pneumonia (Freeman & Leigh, 2023). However, the morphology of the chest CT images is greatly affected by the demographics and epidemiology of society. Some patients had only pure GGO patterns, solid nodules, consolidations, or fibrous stripes. Other patients had more than one of the four morphological lesions that were used to figure out their clinical symptoms (Niu et al., 2021). The way CT scans are used to diagnose viral pneumonia has changed a lot since the invention of the thin-section CT scan, which can find problems in the parenchyma and help confirm the diagnosis of viral pneumonia when a CXR gives false negative results. (Freeman & Leigh, 2023). The viral pneumonias that have the closest similarity in clinical features are SARS and MERS (Park et al., 2020).

2.2 Convolutional Neural Networks (CNN)

CNNs are designed to mimic the structure and function of the human visual system, which is known to process visual information hierarchically, from simple features such as edges and corners to more complex ones such as shapes, objects, and scenes (Zhang et al., 2021) (Jozwik et al., 2017). CNN architecture is based on learning how to represent features in a hierarchical way from raw input data like text, images, or speech signals (Alzubaidi et al., 2021). To do this, CNNs use many convolutional layers, which are learnable filters that mix input signals with a group of learned kernels to pull out local features at various levels of abstraction (Li et al., 2019). Then, these features are put through a set of nonlinear activation functions, which are known as activation layers. These add nonlinearity to the neural network and let it learn the complex and nonlinear mapping between the input and output (Alzubaidi et al., 2021). Finally, the output of the last layer, typically a fully connected layer or a softmax layer, is used to make a prediction or classification of the input data. CNNs' ability to automatically extract features from raw data without requiring manual feature engineering, their scalability to large datasets, and challenging tasks like object recognition, image segmentation, and natural language processing are their main benefits over rudimentary machine learning algorithms

(Alzubaidi et al., 2021). The swift advancement of medicine, in conjunction with diverse computational methodologies, has resulted in a multitude of advancements. Artificial intelligence (AI) is a subdivision of computer science.

The human body images were generated for medical operations and diagnosis in the field of medical imaging. The main emphasis of medical imaging lies in the disciplines of radiology, cardiology, and pathology. A case study of radiology involves the detection of automated microcalcifications and masses on mammography, as well as lung nodules on chest X-rays and CT scans (Le, Yefeng, Gustavo, & Lin, 2017). Computer-aided diagnosis (CAD) has emerged as a supplementary diagnostic tool for experts and doctors. CAD applications are diverse and utilise algorithms to generate hypotheses for detecting anomalies in several organs, including the breasts, lungs, heart, gastrointestinal tract, nervous system, and the entire body (Fujita, 2020). Nevertheless, if the algorithm utilised by the CAD system lacks precision, it could generate incorrect positive results and erroneously diagnose or misidentify the patient as having an abnormality, leading to a different outcome. The emergence of artificial intelligence (AI) and the advancement of convolutional neural networks (CNN) have enabled the capability to recognise images and detect things within them (Sarvamangala & Kulkarni, 2022).

The creation of CAD systems requires a significant investment of time and effort. Thus, humans are utilising artificial intelligence (AI) capabilities to enhance the effectiveness of CAD systems (Leeuwan et al., 2022). The CAD algorithm is comprised of two distinct components: initial lesion diagnosis and false-positive reduction. The initial step involves identifying the primary lesion by pre-processing, segmenting the body areas, generating potential candidates, and extracting relevant features. On the other hand, the visual representation of CAD findings by the radiologist constitutes a false-positive reduction (Liang et al., 2021). The elucidation of CAD systems has proven to be arduous. After acquiring the medical data and reference standards, it is necessary to annotate the data (Summer, 2017). An expert is necessary to precisely identify the specific location of the abnormality. For optimal accuracy in determining the location of the lesion, it is advisable to have multiple specialists annotate the photos. Therefore, the most reliable information is derived from extensive datasets, which are typically inaccessible until a well-financed study.

The introduction of deep learning aimed to enhance neural network architectures by incorporating multiple layers to enable the representation of complex abstractions. The application of DL effectively identified objects in real-world photos and acquired their characteristics from the training data. Therefore, certain researchers have discovered that computer-aided design (CAD) methods that necessitate the human selection and annotation of hand-chosen parameters and hand-crafted features are expensive, time-consuming, delicate, and not dependable when used with unfamiliar data (Summer, 2017). Deep learning obviates the need for laborious manual hand-tuning techniques.

The utilisation of chest X-rays (CXR) and computer tomography (CT) images has demonstrated their significant value in extracting COVID-19 features through the application of transfer learning (Apostolopoulos & Mpesiana, 2020). The new network incorporates VGG19, MobilenetV2, Inception, Xception, and ResNet v2, to execute classification tasks. VGG19 demonstrated the highest accuracy in binary classification, obtaining a rate of 98.75%. In multiclass classification, VGG19 achieved an accuracy of 93.48% (Tulin et al., 2020). The DarkCovidNet, which is built upon the DarkNet architecture, achieved a binary classification accuracy of 98.8% and a multiclass classification accuracy of 87.02%. This was accomplished by utilising two datasets and employing feature extraction and transfer-learning techniques. Various CNN architectures such as VGG, Inception, ResNet, NASNet, Xception, MobileNet, and

DenseNet were selected and yielded exceptional results on the ImageNet dataset (Ohata et al., 2021). Subsequently, they opted for various arrangements from the learned CNN architectures, eliminated the fully connected layers from these arrangements, and retained the convolutional and pooling layers. It is essential to transform the CNN architectures into feature extractors for CXR images by utilising the sub-datasets as input.

The ultimate categorization employed machine learning algorithms including Support Vector Machine (SVM), K-Nearest Neighbour (KNN), Bayes, Random Forest (RF), and Multilayer Perceptron (MLP). The author elucidated the several categories of classifiers: KNN is an instance-based algorithm, RF is a method that utilises decision trees, MLP is based on neural networks, and there are more methods as well. The MobileNet-SVM architecture achieved a superior accuracy rate of 98.5% for the initial dataset, while the DenseNet201-MLP model attained the highest accuracy rate of 95.6% for the subsequent dataset. Through logical inference rules, the computer may automatically reason about the arguments in these formal languages, which has been the goal of several AI programs that have aimed to hard-code information about the world. An ML logistic regression system suggests that a caesarean section is necessary for birth.

Naive Bayes is another easy machine-learning technique for distinguishing between spam and legitimate email (Goodfellow et al., 2016). The data representation dramatically affects the effectiveness of simple ML techniques. For example, the obstetrician has to directly feed recommended information to the machine learning algorithm to inform the doctor of relevant information correctly. Each piece of information fed into the algorithm is called a feature. The application of logistic regression to the algorithm allowed the learning of each feature representation to correlate with various outcomes. However, the ML algorithm cannot produce accurate predictions if the feature is in a format other than the one declared appropriate for the algorithm. It is also tricky for ML to identify the features and extract them accurately, especially when the feature is in pixel format. For example, an ML algorithm was written to identify a truck on the road. A house has gates and fences, which are highlighted as essential to identify the main gate. Although describing a gate in pixel values is problematic because it is not just the simple geometric shape but also the shadows of an object falling on another item, the sunlight reflected from the shiny metal surfaces, the fireguard at the foreground obscuring part of the object that happens to be in the background, and many other obstacles. An image of a red truck taken at night may have nearly dark pixels, and depending on the angle from which the image is taken, the truck's silhouette may take various shapes. One of the solutions is representation learning, which allows the system to quickly adapt to new jobs with little human intervention. An example of representation learning is the autoencoder associations with an encoder function that converts the input data into various depictions of the input data and a decoder that alters new representations back into the original format (Goodfellow et al., 2016).

Deep learning addresses this fundamental issue in representation learning; deep learning introduces representations in terms of other, more straightforward representations. The ability to generate complicated concepts out of smaller ones is a feature of deep learning. Multilayer Perceptron (MLP) is a feedforward deep-learning model with a mathematical function mapping the set of input data to output data, allowing the ML to learn the proper representation of the data, and the depth allows the ML to learn layers of representation step-by-step (Goodfellow et al., 2016).

2.2.1 Deep Learning (DL) and CNN Architectures

Deep learning is a subfield of AI and ML; it is considered a key technology of the current Fourth Industrial Revolution (Sarker, 2021). The capacity of DL to learn by feeding data to the models has made it an important issue to research and develop (Sarker, 2021). The most renowned deep learning framework is CNN architectures, which implement various approaches from computer vision to natural language processing (Kumar, 2021). The CNN architecture can learn enormous amounts of data and offer excellent results on many intricate cognitive tasks, similar to or even exceeding professional human performance expectations (Alzubaidi et al., 2021). DL has outperformed renowned ML methods in many fields, such as cybersecurity, NLP, bioinformatics, robotics and control, medical imaging, processing of medical data, and several more (Alzubaidi et al., 2021). Researchers are inspired to apply the DL algorithm to extract discriminative features from representative information with the least manual labour and specialised expertise (LeCun et al., Deep learning, 2015). The multitiered data architecture retrieves the low-level element in the first layer and the high-layer component in the last layer (Alzubaidi et al., 2021). The approach was initially adopted through AI, which simulates the operation of the sensory core of the human brain. The human brain can extract data from a different environment and identify it immediately. More accurately, the human brain can process many inputs, depending on the objective, classifying the object as the immediate output. CNN applies automatic detection to detect substantial features without human oversight, alleviating the burden and saving time, so it is the most widely used technique, especially for the classification and detection of images (Alzubaidi et al., 2021).

DL also had its drawbacks, including lack of training data, imbalanced data between classes, interpretability, uncertainty scaling, catastrophic interference, model compression, overfitting, underfitting, disappearing gradient problems, immense multiplication of gradient problems and underspecification (Alzubaidi et al., 2021).

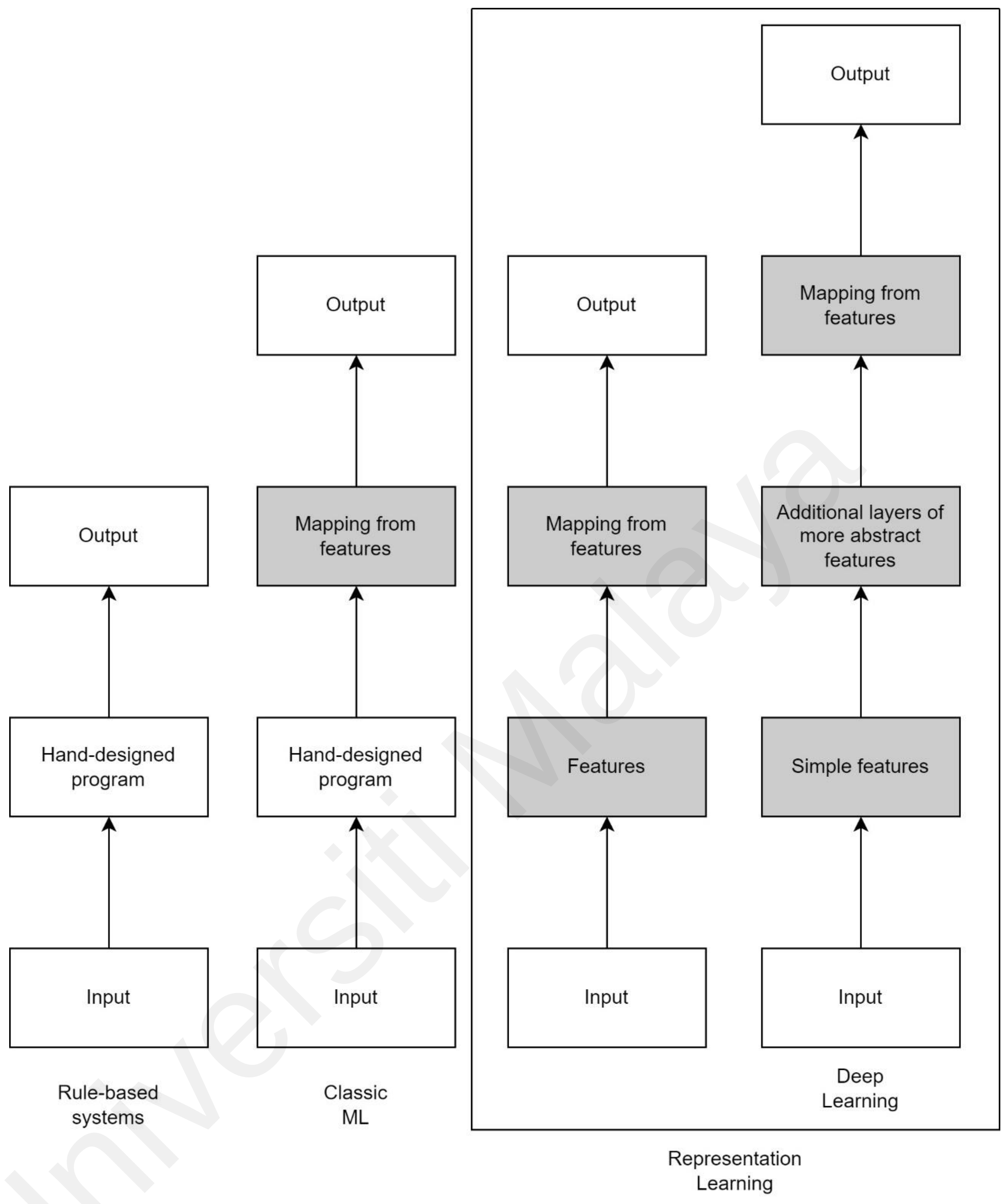


Figure 2.2: Types of AI Discipline (Goodfellow et al., 2016)

Figure 2.2 describes the types of AI discipline and the evolution of AI, from ML to DL (Goodfellow et al., 2016). The availability of training data increases, and the advancement of computer infrastructure on both hardware and software encourages deep learning techniques to evolve and become more valuable (Sarker, 2021). The design of DL begins by learning the intricate structures of deep learning neural networks by applying practical mathematics like linear algebra, probability, and information theory, the fundamentals of DL and ML, mathematical computation, deep feedforward networks, contemporary mathematical techniques, and research (Goodfellow et al., 2016; Sarker, 2021). With the discovery of the AlexNet model in 2012 and the High-Resolution model in 2020, CNN architectures have advanced exponentially. International corporations like Google, Microsoft, AT&T NEC, and Facebook recognised the potential of CNN and expanded their research and development in the field (Celine & Kumar, 2021).

2.2.2 Deep Feedforward Networks

Multilayer perceptrons (MLPs) and deep feedforward networks are the same (Gupta & Raza, 2020). Modern DL's primary goal is to solve real-world problems. DL technologies are widely used in a variety of industries (Alzubaidi et al., 2021; Sarker, 2021). Advanced DL requires a state-of-the-art framework for supervised learning, and the intricate neural network becomes more complex as layers are added to create depth (Hu et al., 2021). It is easy for a person to perform simple tasks and activities like manually mapping representations of information from input to output (Sarker, 2021). However, carrying out intricate and complex operations that cannot be defined or classified by associating one vector with another in large quantities is more complicated, tedious, and time-consuming.

The structure of MLPs is reasonably straightforward. The input data travels through the network channel, is assessed through transitional computations, and, finally, is directed to the output (Goodfellow et al., 2016). Feedforward neural networks do not have feedback connections. If a feedforward neural network is extended to include feedback connections, they are recurrent neural networks. Feedforward neural networks are designed with various functions connected to a chain structure to form neural networks. The chain structure comprises many layers, from the first to *n*. The learning algorithm of the MLPs determines by what method to apply these layers to achieve the required output.

In between the layers of the networks, there are hidden layers. Every hidden layer contains vector-valued elements that determine the model's width, and the vector element is comparable to a neuron in the human nervous system. Although the model of the brain inspires modern neural network research, feedforward networks use mathematical probability and engineering disciplines to achieve results. The simplest way to understand an MLP is to start with linear networks and study the limitations of the linear function that it has to overcome (Brownlee, 2022). A linear function can only have an input and an output. Hence, it is difficult for a linear model to understand the interaction that begins with two inputs. A case of MLP commercial applications is CNNs applied for object identification and detection from images (Goodfellow et al., 2016).

Linear models must extend to accommodate nonlinear functions but include a transform input. The transformed input is a bundle of feature representations describing the actual input instead of using the same input (Brownlee, 2022). Linear and logistic regressions are examples of linear models. These regressions are simple to use because they are efficient and reliable. However, due to their simple nature, the regressions have limitations because they comprise linear functions that only limit one input and output at a time (Zhang et al., 2021). Linear and logistic regressions cannot learn and receive two input variables to create an output (Goodfellow et al., 2016).

CNNs are a subfield of deep feedforward networks that are specially developed to handle grid-like topologies like images (Goodfellow et al., 2016). The kernel technique can obtain nonlinear learning as well by utilising transform input. There are several methods to choose the mapping of the transform input into the function. The first method is to apply generic transform input (Brownlee, 2020). The algorithm can fit the training set as long as the dimension is high, but it does not have enough preliminary information to solve advanced problems (Google Developers, 2023). The second method is to engineer the transform input manually. Manually engineering the transform input was the usual approach before deep learning became well-known in the industry; humans needed to put decades of work into each separate task, specialising in different elements, to develop the transferable algorithm (Brownlee, 2020). The third method is to learn the transform input using parameters to learn a broad class of functions instead of searching for the proper function (Brownlee, 2020). The final method is the beginning of deterministic mapping and extension feedforward networks with feedback connections, also known as recurrent neural networks (RNN) (Merkh & Montufar, 2019). Training kernel mapping and learning functions with response over a single vector are some principles that have emerged from the evolution of DL models (Goodfellow et al., 2016). If a set of information representations is described as x, the mapping φ provides a new representation of x. MLPs are the foundation for learning and training deterministic mappings from input to output without feedback. The kernel functions are used on the input representation data to link them into a higher dimension with intricate and complicated relationships to extract the features and provide the necessary output. The higher-dimensional feature mapping means the original input information is linked to new data points that capture non-linear relationships. Mapping information to a higher-dimensional space increases computational costs.

2.2.2.1 Learning XOR

The XOR function is an "exclusive or" operation on two binary values, either 0 or 1. When the two inputs are the same, the XOR function returns null, and vice versa; when the two inputs are different, the XOR function output is one. The concept of XOR functions focuses on simplicity, where the network will train the features to learn four points, and the challenge is to fit the training set (Goodfellow et al., 2016). Linear models have done well so far, but nonlinear functions are more practical in describing the features at a given input using affine transformations controlled by parameters such as activation functions. Instead of using a linear function, XOR problems can be solved by learning representations, which will also help the model generalise the function. The XOR functions learned that multiple inputs in the network cannot be linearly separated and require intricate models like multi-layered models (Goodfellow et al., 2016).

2.2.2.2 Gradient-Based Learning

The training ML and DL models with gradient descent are the foundation for designing and training CNN architectures. Neural networks are non-linear, which is the main distinction between linear models. It comprises computing the gradient of a loss function depending on the applied model's parameters. The parameters are attuned to reduce the loss. The neural networks cause interesting loss functions to become non-linear, usually trained by gradient-based optimisers (Goodfellow et al., 2016).

For MLP, all weights are initialised to small random values, and the biases start at zero or positive values. Firstly, there is the option to apply the cost function, loss function, or objective function using the cross-entropy methodology to train the data and determine the magnitude of the gap between the neural network's predictions and the actual values (Goodfellow et al., 2016). The most straightforward approach is to predict the estimates of the output based on the input. Regularisation strategies like weight decay can be applied to the primary cost function. The modern neural network uses maximum likelihood. The gradient of the objective function must be substantial and consistent to employ as a guide for the learning algorithm; otherwise, the function is saturated, causing the gradient to be flat. The saturation of the gradient descent happens when the activation function is the output of the hidden units. The output unit is directly proportional to the cost function chosen in the neural network. However, the output unit can be hidden in any neural network model. The output units are also called linear units based on affine transformation without linearity. The log-likelihood function is used in ML to evaluate and observe input data. Maximising it is equal to minimising the mean squared error. The output units, or layer, may not necessarily be the last layer. There are other examples of output units. For instance, the output can be linear units, sigmoid units, softmax units, and others, depending on the neural network and the type of distributions, which can guide how to design a suitable cost function to produce the output layer (Goodfellow et al., 2016).

The choice of neural networks trained with gradient-based optimisation determines the type of output units. Another choice is to determine the unit in the hidden layer of the model. Rectified Linear Units (ReLU) is a standard default to apply as a hidden unit. The choice of applying the type of suitable units is usually impossible to predict in advance, and designing the neural network requires a lot of trial, error, and intuition to choose a hidden unit that works well for the neural network (Goodfellow et al., 2016). The performance of the model will be assessed with validation sets. Some hidden units in the neural network model cannot be differentiable, e.g., when the value is 0. Therefore, the gradient descent can perform well in practice even though ReLU is implemented in the model because the neural network training algorithms usually do not reach the cost function's local minimum but merely reduce the value significantly. ReLU is comparable to linear units; the only variance is that half of the ReLU's output domain is 0. Whenever the unit is above 0, the derivatives through ReLU remain large. Logistic sigmoid and hyperbolic tangents were utilised before the introduction of ReLU. The sigmoid method was discouraged because the probability predicts the binary variable, which is 1. The problem with the sigmoid method is that when the unit is positive, it will be saturated at the high value, whereas when the unit is hostile, it will be saturated at the low value.

2.2.3 CNN Hyperparameters

In CNN, there are several parameters applied for image training. The following parameters are standard and can be found in various CNN architectures: i) convolutional filter magnitude; ii) the number of kernels; iii) the number of strides; iv) padding value; v) activation function; vi) pooling; and vii) learning rate.

2.2.3.1 Convolutional Filter Size

The filter, or kernel size, is one of the hyperparameters that must be chosen carefully to ensure the model can adapt the correct information from the data. The kernel's magnitude determines the model's local or global receptive field, the region of the input data that impacts the activation of a particular feature map in the output (Hoogen et al., 2023). The convolutional kernel size refers to the dimension of the filter used to extract features from an input image. Generally, the convolutional filter size is a square matrix with odd dimensions. A smaller filter size, such as 3 x 3, is commonly used in the earlier network layers, allowing the network to capture fine details and edges in the image. The larger filter sizes, such as 5 x 5 or 7 x 7, can be used in deeper network layers to capture more complex features or global features such as shapes and objects.

The choice of filter size also depends on the size of the input image. If the input image is small, it may be beneficial to use smaller filters to prevent the loss of too much information in the convolutional layers. Conversely, larger filters may capture more high-level features if the input image is large (Hossain & Sajib, 2019).

Another important consideration when choosing the filter size is the overlap between adjacent filters. Overlapping filters can help capture more information regarding the arrangement, position, and distribution of the targeted object in the image and prevent loss of information at the edges (Nwe & Lynn, 2019). However, too much overlap can increase computational costs and slow the training process. The input data's exact tasks and features determine the kernel size. It is often determined through trial and error or by using techniques such as grid search to optimise the hyperparameters of the network.

2.2.3.2 Filters

In a CNN, filters are kernels, one of the hyperparameters in the neural network, which are small matrices convolved with the input image to extract features. The number of kernels applied in a CNN architecture can be tuned constantly during training. Every filter produces a separate feature map, representing a specific feature the filter is designed to detect (Goodfellow et al., 2016; Sahoo, 2018). The number of filters used in a layer can affect the capacity of the CNN to identify different characteristics of an input image. A larger number of filters can intensify the CNN’s ability to extract more complex features, while a smaller number of filters can lead to faster training times and a lower chance of overfitting. The depth of the CNN models also impacts the filter size. The more profound CNN architecture requires more filters to detect more intricate characteristics from the input data, while smaller kernel sizes can lead to a faster training time and a reduced possibility of overfitting. The number of filters used in the first few layers of a CNN architecture is usually lower than those used in the later layers because the lower layers of the network detect low-level features of the images, such as edges and corners. The subsequent layers detect more complex features, such as objects and shapes within the images. The number of filters in a CNN plays an essential role in the network’s ability to extract useful features from an image, and it is a hyperparameter that should be carefully tuned during training to optimize the performance of the network.

2.2.3.3 Stride

Stride in CNN designates the number of pixels by which the convolutional filter is shifted across the input image or feature map (Shahid, 2019). It determines the number of steps the pixel has to move in the CNN. The default value for stride is one. It can be tuned to control the spatial downsampling or upsampling of the output feature map. Stride plays a significant role in determining the magnitude of the output feature map. The following formula calculates the size of the output feature map:

Output size = (Input Size – Filter Size)/Stride + 1 **2.1**

For instance, if the input size of the image is 32 x 32, the filter size is 3 x 3, and the stride is 1, the output size will be 30 x 30. If the stride is increased to 2, the output size will be 15 x 15. Increasing the stride decreases the output feature map's spatial resolution, leading to more efficient computation and faster training times. However, this also results in losing information and details in the output feature map. In contrast, decreasing the stride results in a larger output feature map with more spatial resolution but requires more computation and longer training times.

2.2.3.4 Padding

Padding in CNN refers to adding extra zeros or any other predefined value around the input data to ensure that the output feature maps have the exact spatial resolution as the output feature maps, which is an essential hyperparameter in determining the result of the CNN. It processes an image by moving across and scanning the features individually and converting the pixels into data (Perera, 2018). Padding is applied to images because the image will shrink during CNN operations if the CNN model is more profound and comprises many layers (Nanos, 2023). The centre pixels of the image are often involved with numerous convolutional activities throughout the training and validation processes, and the cornel pixels of the images are often neglected. Therefore, the edges of the images will not be relevant because the features were not extracted. There are two main types of padding in CNNs: valid padding and the same padding (Pandey, 2020).

Valid padding is no padding added to the input data. The filter is applied only to the valid part of the input, resulting in an output feature map that is smaller than the input feature map. Valid padding is proper when the input size is large and the model is deep, as it reduces the computational cost and memory requirement (Nanos, 2023).

The same padding applies when the size of the output feature map is kept the same as the input feature map by adding zeros or any other predefined value around the input. The same padding is functional when the model needs to maintain the spatial resolution of the input, especially in shallow networks or when the input size is small. It adds features to the surrounding pixels of the image to ensure the input and output sizes are equal (Nanos, 2023)

Another padding variant applied in one-dimensional convolutional layers is causal padding. The goal of causal padding is to provide information at the beginning of the data that assists in predicting the values earlier (Nanos, 2023).

The decision to include padding in the CNN architecture significantly impacts the performance of a CNN, as it affects the size of the output feature maps, which in turn affects the number of parameters and the computational complexity of the model.

2.2.3.5 Activation Function

The activation function is a vital component of a CNN as it helps introduce nonlinearity into the model, allowing it to handle complex, nonlinear functions between input features and output labels. The activation function applies element-wise after each convolutional or fully connected layer. Some of the most commonly used activation functions in CNNs are ReLU (Rectified Linear Unit), Leaky ReLU, Sigmoid, Tanh, and Softmax (Himanshu, 2019).

2.2.3.6 ReLU

ReLU (Rectified Linear Unit) is a commonly used activation function in convolutional neural networks (CNNs). ReLU works similarly to a linear function that provides outputs if the input is positive, or else the output will be zero. The mathematical expression for ReLU is:

$$\varepsilon f(x) = \max(0, x)\varepsilon \qquad 2.2$$

Where x is the input to the function and f(x) is the output.

ReLU has several advantages as an activation function in CNNs; it has a non-linear function that allows the network to model complex relationships between the inputs and outputs. The sparsity that ReLU produces in the activation maps means that many neurons in a layer output zero, which aids the network in decreasing the computational intricacy and prevents overfitting. ReLU only involves simple thresholding, so it is a computationally efficient and straightforward function that is easy to implement and understand.

ReLU is an unbounded activation function, which means that the output can grow without limit as the input becomes bigger, leading to numerical instability and making the network difficult to optimise. However, ReLU has limitations. For instance, if the output is zero for all inputs, the ReLU neurons can "die". Dead neurons in ReLU can occur during the training of the neural network if the weights in the algorithm are updated so that the output is always negative. Once a neuron is "dead", it will continuously output zero for all future inputs, and the gradient of the loss concerning its weights will also become zero. Dead neurons reduce the training process, making learning difficult (Brownlee, 2019).

The limitations of ReLU were addressed by proposing several ReLU variants, like Leaky ReLU, Parametric ReLU, and Exponential ReLU. These variants introduce slight modifications to the original ReLU function to address the limitation above while maintaining the advantages of ReLU.

2.2.3.7 Leaky ReLU

Leaky ReLU is a type of activation function derived from the popular ReLU that addresses its limitations. The Leaky ReLU function is:

$$\boldsymbol{f(x) = \max(ax, x)} \qquad \textbf{2.3}$$

Where *x* is the input to the function, *a* is a small positive constant called the "leakiness" parameter, and max is the maximum function that returns the higher value between its two inputs. The leakiness parameter is usually assigned to a small value like 0.0, meaning the function has a slope of 1 for positive and 0.01 for negative inputs. The small slope exists for negative inputs to allow the Leaky ReLU function to avoid the "dying ReLU" problem, which happens when the ReLU function outputs 0 for all negative inputs, causing the gradient to be 0, preventing further learning (Himanshu, 2019).

Leaky ReLU has become famous for many deep learning applications, especially in computer vision and natural language processing tasks. One of the main advantages of Leaky ReLU over other activation functions is its ability to handle negative input values without causing saturation, which can occur with other activation functions such as the sigmoid and hyperbolic tangent functions, improving the performance of deep neural networks. However, it is essential to note that the optimal choice of activation function depends on the specific task and dataset, and experimentation is necessary to obtain the best option.

2.2.3.8 Sigmoid

Sigmoid is applied as an activation function in a neural network; it is a nonlinear activation function that takes any real-valued number and "squashes" it into a range between 0 and 1 (Goodfellow et al., 2016). The definition of the sigmoid function is:

$$\boldsymbol{sigmoid(x) = \frac{1}{(1+e^{(-x)})}} \qquad \textbf{2.4}$$

Where x is the input to the function and e is the exponential function.

The sigmoid function is nonlinear. The output of the sigmoid function is always between 0 and 1, which is helpful for binary classification tasks where the goal is to predict a probability that an input belongs to one of the two classes. It is also a smooth function, meaning it has a continuous first derivative and is helpful for gradient-based optimisation algorithms like stochastic gradient descent (SGD).

The disadvantage of the sigmoid function is the vanishing gradient, which causes difficulty when training deep neural networks. When the input to the sigmoid function is very large or small, the gradient of the function becomes very small, causing difficulty in updating the network weights during training. It can be alleviated to some extent by using the ReLU activation function or its variants, like ReLU.

2.2.3.9 Tanh (Hyperbolic Tangent)

Tanh is the hyperbolic tangent, a popular activation function used in neural networks that maps its input to a range between -1 and 1. The formula of the hyperbolic function is:

$$\mathbf{tanh}(\boldsymbol{x}) = \frac{(e^{x}-e^{-x})}{(e^{x}+e^{-x})} \quad , \qquad \mathbf{2.5}$$

where e is Euler's number and x is the input value.

Tanh is a smooth, S-shaped, zero-centred function, meaning its outputs are centred around zero. It is commonly used in hidden layers of neural networks and is preferred over the sigmoid function because it can produce both positive and negative output values. Tanh's property allows the model to process more complex neural networks with nonlinear relationships in the data. Tanh also suffers from the vanishing gradient problem. When the input values to the Tanh function are significant, its derivatives approach zero, which makes it difficult for the network to learn from the data.

2.2.3.10 Softmax

Softmax is an activation function commonly used in the output layer of a neural network to produce a probability distribution over multiple classes. It is beneficial for multi-class classification problems, where the task is to predict the probability of each class. The Softmax function takes a vector of arbitrary real-valued scores as input and produces a vector of the same size as output, where each element represents the probability of the corresponding class. The output vector is a probability distribution, meaning that the sum of all its elements equals one (1).

The formula for the Softmax function is:

$$\boldsymbol{\sigma(z_j) = \frac{e^{z_j}}{\xi_{k=1}^{K} e^{z_k}}}, \qquad \textbf{2.6}$$

where *z* is the input vector of scores, $\sigma(z_j)$ is the output vector of probabilities, and *K* is the number of classes.

In this formula, e^{z_j} represents the “importance” of the *j*-th class, and the denominator $\xi_{k=1}^{K} e^{z_k}$ is the sum of the “importance” values of all classes. The softmax function normalises these “importance” values by dividing each by the sum so that they add up to one, which makes them suitable to represent probabilities, which is easy to interpret. The formula is differentiable, which makes it suitable for use in backpropagation algorithms to train neural networks. The softmax function is also monotonic, which means that increasing the score for one class will increase the probability of that class and decrease the probability of other classes.

The activation function option depends on the specific tasks of the assigned CNN architecture. Trial and error is often required to determine which activation function works best for the given problem.

2.3 Types of CNN Architecture

The term architecture applies to the overall structure of the network, including the number of units they should have and how they are connected. Neural networks are arranged into groups of units known as layers. Each layer will consist of weight, bias, and units. The architecture design considers the depth and width of each layer. The application of CNN has proven useful, especially when human experts are limited and unavailable to perform the tasks personally (Yadav & Jadhav, 2019). Sometimes, experts or highly skilled professionals cannot explain the decisions made. This situation can be observed in understanding languages and medical decisions like reading CT or CXR images to differentiate between COVID-19, viral and bacterial pneumonia, and speech recognition. Problems and solutions sometimes update over time, like price prediction, stock preference, weather prediction, and tracking. (Carvelli et al., 2020; Fauw et al., 2018; Topol, 2019; Kermany et al., 2018). Table 2.1 shows the prominent CNN architecture from 1998 to 2021.

Table 2.1: Prominent CNN Architectures from 1998 – 2021.

Sources	CNN	Contributions
(Wu et al., 2021)	AutoFormer (2021)	Microsoft Research Asia once again used transformer-based architecture that uses a "masked token prediction" pretraining task to learn better representations of images.
(Chen et al., 2021)	CrossViT (2021)	Microsoft Research Asia combines the transformer-based architecture of ViT with cross-attention mechanisms to capture global and local features in images.

Table 2.1: Continued.

(Liu et al., 2021)	Swin Transformer (2021)	Microsoft Research Asia introduced a transformer-based neural network that uses hierarchical feature pooling to capture images' local and global spatial relationships.
(Zhang et al., 2020)	ResNeSt (2020)	The Chinese University of Hong Kong builds this CNN upon the residual learning framework of ResNet and introduces a new “split attention”.
(Dosovitskiy et al., 2020)	Vision Transformer (ViT) (2020)	Google introduces it as a transformed-based neural network that uses a self-attention mechanism to capture spatial relationships between image patches.
(Radosavovic et al., 2020)	RegNet (2020)	It uses the ‘regularisation’ module to encourage the network to learn more efficiently.
(Tan & Le, 2019)	EfficientNet (2019)	It uses a compound scaling method to optimise network depth, width and resolution.
(Howard et al., 2017)	MobileNet (2017)	The model is optimised for deployment on mobile and embedded devices
(Hu et al., 2017)	SENet (2017)	It was introduced to improve the ability of the network to model interdependencies between feature channels.
(Huang et al., 2017)	DenseNet (2017)	It concatenates feature maps from all previous layers.
(Xie et al., 2017)	ResNeXt (2017)	The improved version of ResNet with ‘cardinality’ increases the model's width instead of the depth.
(Szegedy et al., 2016)	Inception v4 (2016)	It is an improved version of the Inception module

Table 2.1: Continued.

(He et al., 2015)	ResNet (2015)	It introduces residual learning to train intense networks.
(Szegedy et al., 2015)	Inception v3 (2015)	It is an improved version of Google.
(Szegedy et al., 2014)	GoogleNet (2014)	Uses 'inception' modules to capture multiple scales of features
(Simonyan & Zisserman, 2015)	VGG (2014)	It consists of tiny filters and deeper layers than AlexNet.
(Zeller & Fergus, 2013)	ZFNet (2013)	It is a modified version of AlexNet with smaller filters.
(Krizhevsky et al., 2012)	AlexNet (2012)	Winner of ImageNet Large Scale Visual Recognition 2012
(LeCun et al., 1998)	LeNet (1998)	CNN architecture, used for handwritten digit recognition

2.3.1.1 LeNet

LeNet, formerly known as LeNet-5, is a CNN architecture that Yann LeCun and his colleagues at AT&T Bell Laboratories introduced in 1998 (LeCun et al., 1998). LeNet is a simple architecture and one of the first to be developed. It was designed for character recognition applications, specifically for recognising handwritten digits. It is an old-school CNN architecture that cannot be scaled to all image classes (Alzubaidi et al., 2021). However, some research has applied novel LeNet to CT images (Islam & Matin, 2020). LeNet has two convolution layers, two pooling layers, and three fully connected layers, making it seven. The input layer to the network is grayscaled images, and the output probability distribution over ten classes corresponds to the digits 0–9. Le Net served as the pioneer of CNN architecture in deep learning. Although it is no longer widely used for image classification, it is the base for many modern CNN architectures (Alzubaidi et al., 2021).

LeNet-5 comprises several convolutional and pooling layers, followed by one or more fully connected layers with Gaussian connections, and, finally, the output layer. The convolutional layers use small 5 x 5 or 3 x 3 kernels and a small number of channels, decreasing the parameters in the model to avoid overfitting. The pooling layers are used to reduce the spatial dimensions of the feature maps, and max pooling or average pooling is used to extract the most prominent features.

One of the critical contributions of LeNet was the application of the "gradient-based learning" algorithm for training neural networks, also known as backpropagation (LeCun et al., 1998), which used the chain rule of calculus to compute the gradients of the loss function efficiently concerning the model parameters. Backpropagation allows the network to learn from the vast amount of labelled data and improve its performance over time.

LeNet achieved advanced performance in several character recognition tasks and has since inspired the development of many other CNN architectures for computer vision tasks. While LeNet is relatively simple compared to the more recent CNN architectures, it was a pioneering work that helped establish the foundations of deep learning and demonstrated the potential of CNNs for image recognition tasks. Figure 2.3 describes the LeNet diagram.

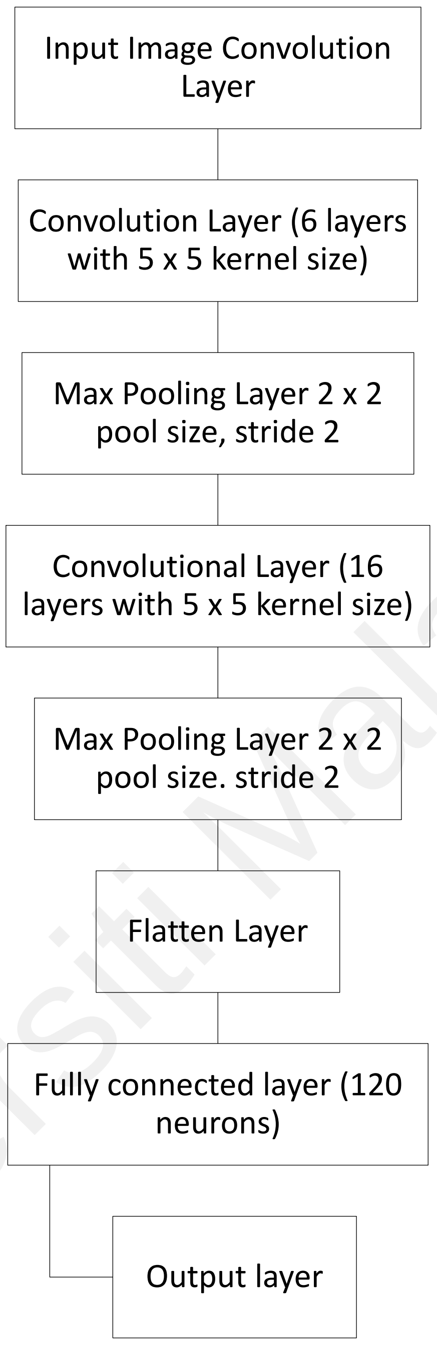

Input Image Convolution Layer
Convolution Layer (6 layers with 5 x 5 kernel size)
Max Pooling Layer 2 x 2 pool size, stride 2
Convolutional Layer (16 layers with 5 x 5 kernel size)
Max Pooling Layer 2 x 2 pool size. stride 2
Flatten Layer
Fully connected layer (120 neurons)
Output layer

Figure 2.3: LeNet Diagram (LeCun et al., 1998)

2.3.1.2 AlexNet

In the 2012 ImageNet competition (Krizhevsky et al., 2012), AlexNet was discovered as an effective neural network applied to various computer vision applications (Szegedy et al., 2015). AlexNet pioneered CNN in computer vision and set standards for the neural network. It was introduced to improve learning capabilities by increasing the layers' depth and optimising parameters (Alzubaidi et al., 2021; Krizhevsky et al., 2012). AlexNet recommended ReLU nonlinearity as the activation function, overlapping pooling, dropout, and the effectiveness of DCNN in image recognition. The implementation of ReLU has become the standard technique in CNN training. Since then, convolutional neural networks' quality has significantly improved by using deeper and broader network designs to yield high performance.

AlexNet, though revolutionary when it was discovered, has several limitations, which include a lack of scalability, large numbers of parameters, overfitting, fixed input size, and lack of depth (Ayyar, 2020). AlexNet has 60 million parameters (Krizhevsky et al., 2012), but it is not deep enough to classify intricate features from an image. CT images are complex images with various resolutions, depending on the type of CT equipment used on the patient. CT images differ from natural images, and data preprocessing is required to adjust the image to fit the model. The input size of 224 x 224 is fixed and must be adjusted to fit the model (Krizhevsky et al., 2012). Figure 2.4 shows the architecture of AlexNet.

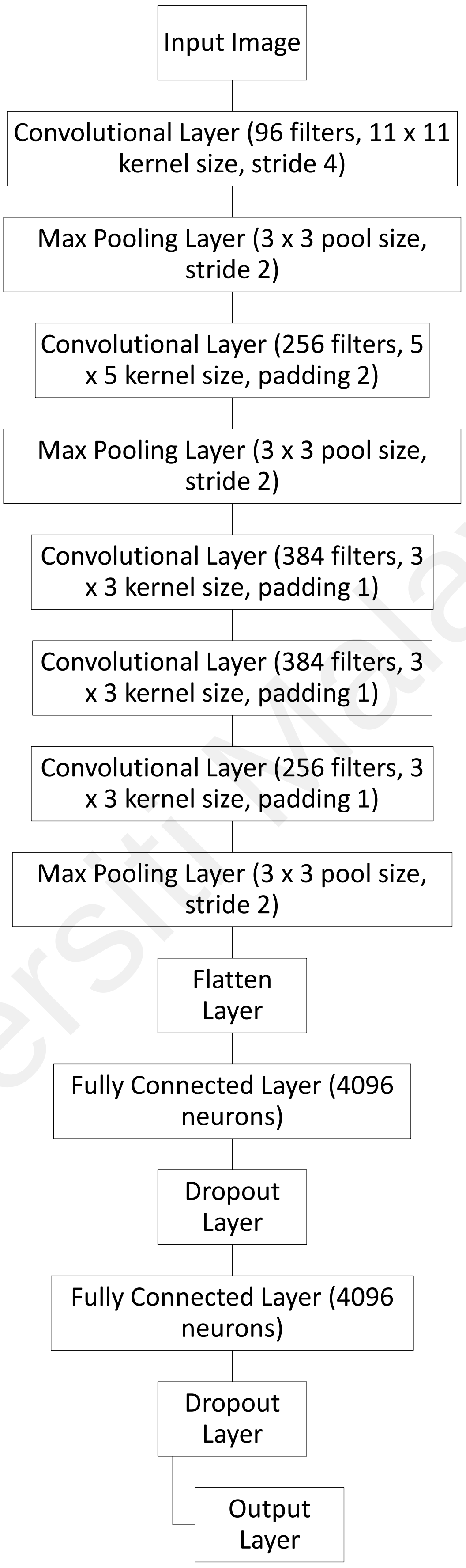

Input Image
Convolutional Layer (96 filters, 11 x 11 kernel size, stride 4)
Max Pooling Layer (3 x 3 pool size, stride 2)
Convolutional Layer (256 filters, 5 x 5 kernel size, padding 2)
Max Pooling Layer (3 x 3 pool size, stride 2)
Convolutional Layer (384 filters, 3 x 3 kernel size, padding 1)
Convolutional Layer (384 filters, 3 x 3 kernel size, padding 1)
Convolutional Layer (256 filters, 3 x 3 kernel size, padding 1)
Max Pooling Layer (3 x 3 pool size, stride 2)
Flatten Layer
Fully Connected Layer (4096 neurons)
Dropout Layer
Fully Connected Layer (4096 neurons)
Dropout Layer
Output Layer

Figure 2.4: AlexNet Diagram (Krizhevsky et al., 2012)

2.3.1.3 ZF Net

ZF Net, also known as "Zeiler and Fergus Net," is a CNN architecture introduced in 2013 by researchers at New York University and Facebook AI Research (Zeller & Fergus, 2013). It was designed to improve the performance of CNN architecture on the ImageNet dataset, a massive visual recognition competition dataset that contains millions of labelled images across thousands of object categories.

AlexNet, which had acquired advanced results on the ImageNet dataset in 2012, served as an inspiration for ZF Net. It retained the overall structure of AlexNet but utilised smaller filter sizes in the convolutional layers to capture more detailed features of the input images. ZFNet also applied smaller stride sizes in the pooling layers, preserving more spatial information in the output feature maps (Zeller & Fergus, 2013).

ZF Net key contribution was applying "deconvolutional" layers to visualise the learned features in the network. Deconvolutional layers are the inverse of convolutional layers and are used to reconstruct the input image from the output feature maps of the CNN layer. Using this technique, the researchers could understand how the network represented the different objects and textures within the input images. Since then, ZF Net has inspired researchers to develop many other CNN architectures that use smaller filter sizes and stride sizes to capture more detailed features and preserve more spatial information. Figure 2.5 describes the layout of ZFNet.

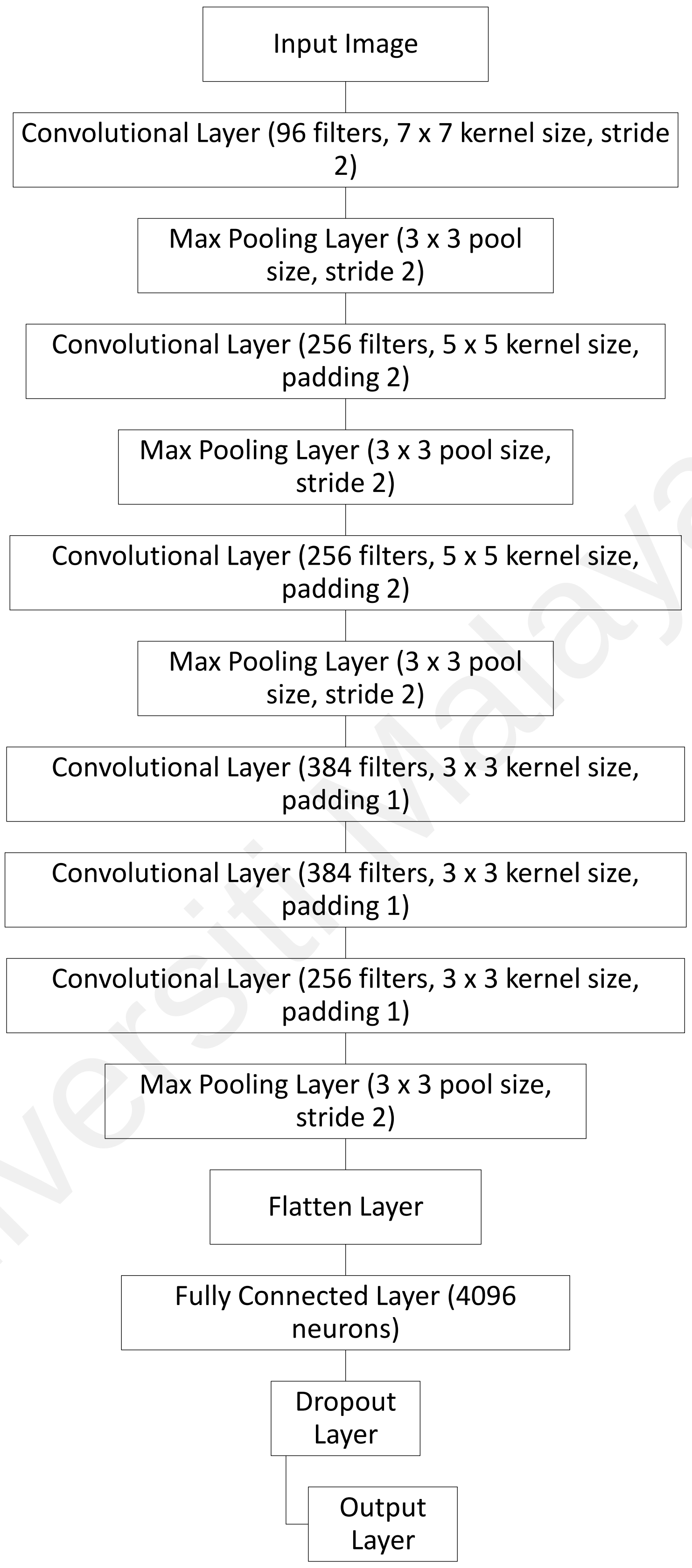

Input Image
Convolutional Layer (96 filters, 7 x 7 kernel size, stride 2)
Max Pooling Layer (3 x 3 pool size, stride 2)
Convolutional Layer (256 filters, 5 x 5 kernel size, padding 2)
Max Pooling Layer (3 x 3 pool size, stride 2)
Convolutional Layer (256 filters, 5 x 5 kernel size, padding 2)
Max Pooling Layer (3 x 3 pool size, stride 2)
Convolutional Layer (384 filters, 3 x 3 kernel size, padding 1)
Convolutional Layer (384 filters, 3 x 3 kernel size, padding 1)
Convolutional Layer (256 filters, 3 x 3 kernel size, padding 1)
Max Pooling Layer (3 x 3 pool size, stride 2)
Flatten Layer
Fully Connected Layer (4096 neurons)
Dropout Layer
Output Layer

Figure 2.5: ZFNet Diagram (Zeller & Fergus, 2013)

Overall, the ZF Net architecture is similar to AlexNet but with slightly different sizes and strides that were optimised through a process called "visualizing and understanding convolutional neural networks" (Zeller & Fergus, 2013), which involved using visualization techniques to understand the behaviour of the model better and improve its performances.

2.3.1.4 VGG Net

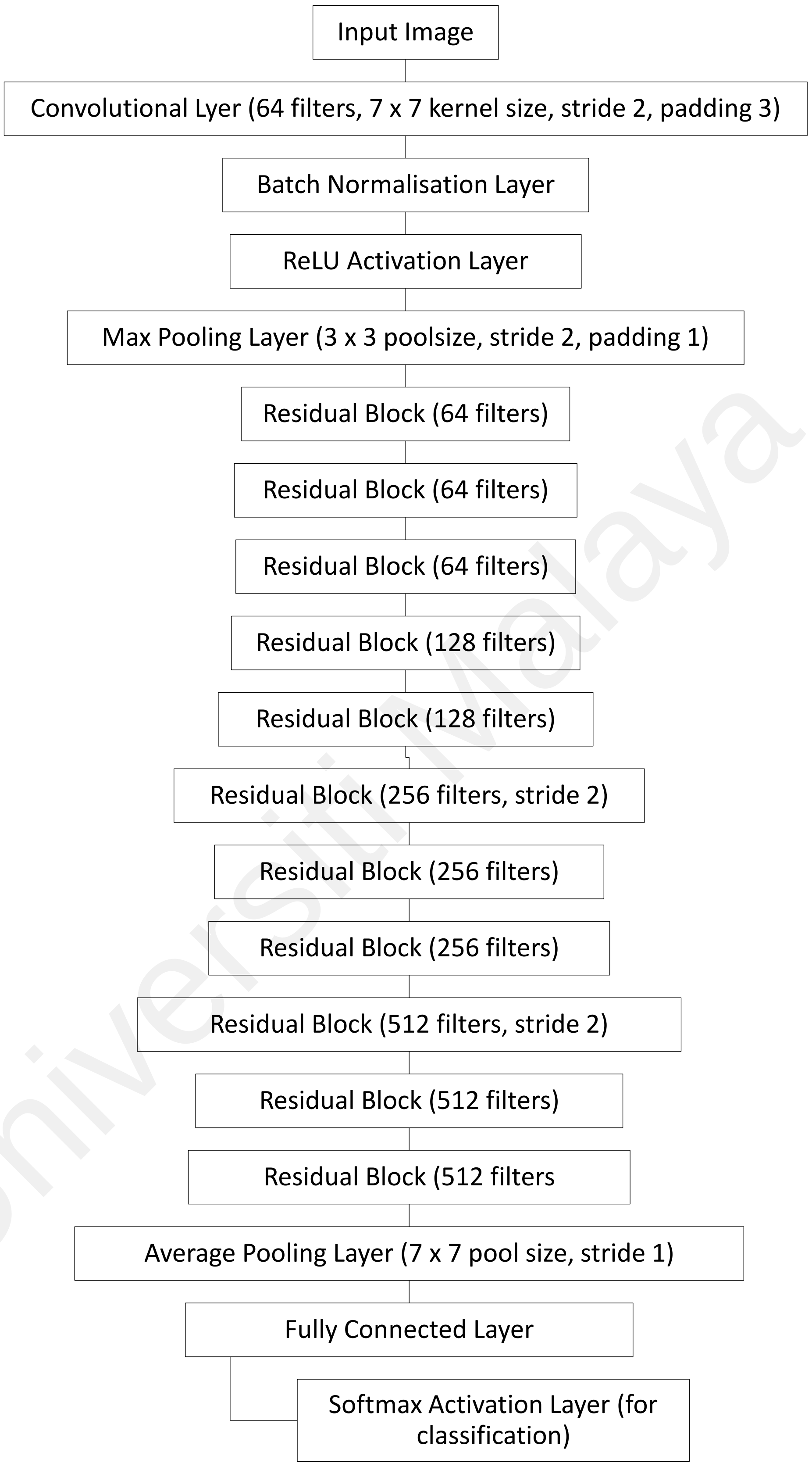
The Visual Geometry Group (VGG) was created to increase the depth of the neural network to increase the model performance (Simonyan & Zisserman, 2015). The deep CNN architecture shown in VGG16/VGG19 has 16 and 19 layers, and uses small convolution filter sizes (3 x 3). As a result, it works better (Simonyan & Zisserman, 2015). The design of VGG is straightforward. The structure of the models applied recurring convolutional layers and max pooling layers to make the model deeper (Simonyan & Zisserman, 2015). The model has small convolutional kernel sizes. Although VGG has compelling features and a simple architectural design, the computational cost of evaluating the network is very high. VGG employs three (3) times more parameters than AlexNet's 60 million parameters, which makes VGG less desirable.

2.3.1.5 GoogLeNet

GoogLeNet, also known as Inception-v1, was developed using only 5 million parameters and a sparse structure instead of fully connected network architectures, drastically cutting the computational cost while outperforming its forerunner. Due to its low computing cost and limited memory, Inception is now practical for big-data settings. The general design principle of Inception is to avoid representational bottlenecks at the beginning of the network.

2.3.1.6 ResNet

ResNet is short for “Residual Network”. Researchers from Microsoft Research first introduced it in 2015. The CNN architecture was developed to address the vanishing gradients issue in AlexNet, VGG, and GoogLeNet (Alzubaidi et al., 2021). The ResNet architecture comprises a series of convolutional, batch normalisation, ReLU layers, and residual blocks. The basic building block of ResNet is called the “Residual Block”. It consists of two convolutional layers with a skip connection or residual mappings that directly create a channel from the input to the output of the second convolutional layer. The “Residual Block” learns residual mappings. In other words, the network learns to model the difference between the input and output of a block instead of trying to learn the output from the input directly, making the network much more profound than traditional neural networks while maintaining good performance. CNN architectures before ResNets applied standard feedforward neural networks where the output of one layer is the sequential input to the next (He et al., 2015). Unfortunately, as the input images become more intricate than standard images, it requires a deeper network to learn and train, causing the gradient's value to become small until the point of insignificance, leading to vanishing gradients (He et al., 2015). ResNet was created to fix vanishing gradients by collecting the leftover data needed to make the output and using those leftover connections to let gradients flow easily (He et al., 2015). Residual blocks were developed to train intense networks. Figure 2.6 is the diagram of a general ResNet model.



Input Image
Convolutional Lyer (64 filters, 7 x 7 kernel size, stride 2, padding 3)
Batch Normalisation Layer
ReLU Activation Layer
Max Pooling Layer (3 x 3 poolsize, stride 2, padding 1)
Residual Block (64 filters)
Residual Block (64 filters)
Residual Block (64 filters)
Residual Block (128 filters)
Residual Block (128 filters)
Residual Block (256 filters, stride 2)
Residual Block (256 filters)
Residual Block (256 filters)
Residual Block (512 filters, stride 2)
Residual Block (512 filters)
Residual Block (512 filters
Average Pooling Layer (7 x 7 pool size, stride 1)
Fully Connected Layer
Softmax Activation Layer (for classification)

Figure 2.6: ResNet Diagram (He et al., 2015)

A residual block consists of two convolutional layers with the same number of filters, followed by batch normalisation layers and ReLU activation layers. The architecture includes 5 sets of residual blocks, with the number of blocks per set depending on the specific ResNet variant. The residual blocks are repeated throughout the network, with the number of filters increasing as the spatial dimensions of the feature maps decrease (He et al., 2015).

2.3.1.7 Inception v3 and Inception v4

Inception v3 was introduced in 2015 after ResNet. It was designed to improve the original Inception network by addressing its limitations, like its high computational cost and many parameters in the network (Szegedy et al., 2015). The novelty of Inception v3 is the use of "factorised" convolutions, which decompose a standard convolution into two smaller convolutions (Szegedy et al., 2015). The "factorised" method decreases the number of parameters and improves computational efficiency. The neural network also uses a "bottleneck" that reduces the number of channels in the input before applying convolutions to reduce computational costs.

Inception v3 consists of Inception module blocks of convolutional layers that use various kernel sizes in parallel. Several kernel sizes capture different feature types in the input image and improve the network's performance. The network can capture a wider range of features with multiple kernel sizes in parallel.

The architecture includes a final pooling layer and a fully connected layer, followed by a softmax output layer for classification. The CNN applied a combination of batch normalisation, dropout, and auxiliary classifiers to improve training stability and prevent overfitting. Figure 2.7 describes the layout for Inception v-3.

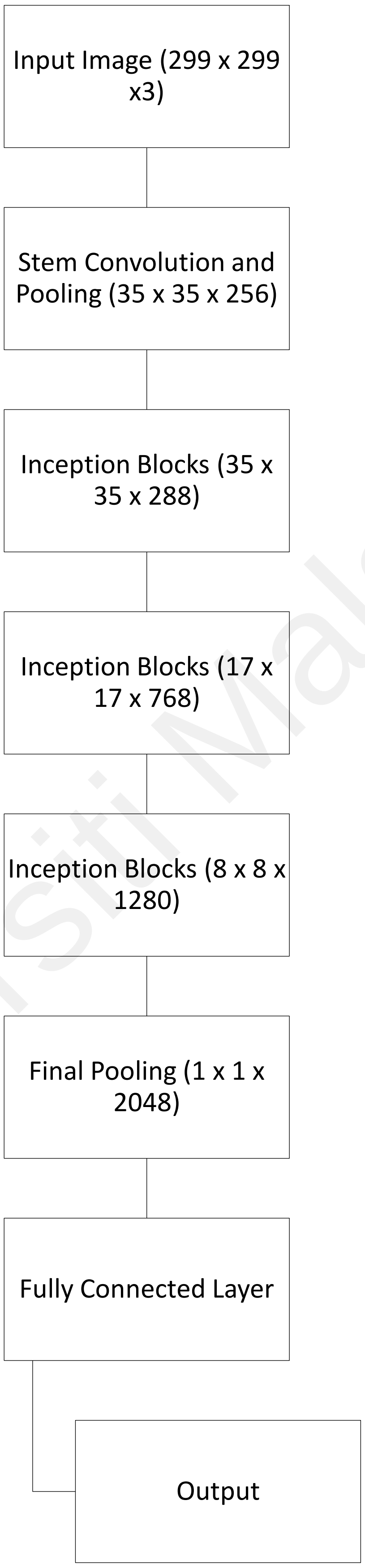

Input Image (299 x 299 x3)
Stem Convolution and Pooling (35 x 35 x 256)
Inception Blocks (35 x 35 x 288)
Inception Blocks (17 x 17 x 768)
Inception Blocks (8 x 8 x 1280)
Final Pooling (1 x 1 x 2048)
Fully Connected Layer
Output

Figure 2.7: Inception v-3 Diagram (Szegedy et al., 2015)

A stem convolution and pooling layer comes first in the Inception v3 architecture, followed by several inception blocks (Szegedy et al., 2015). The Inception blocks comprise multiple parallel convolutional layers with different kernel sizes and stride values concatenated at the end. The output layer of the Inception block is through a final pooling layer, a fully connected layer, and an output layer with 1000 classes for image classification tasks.

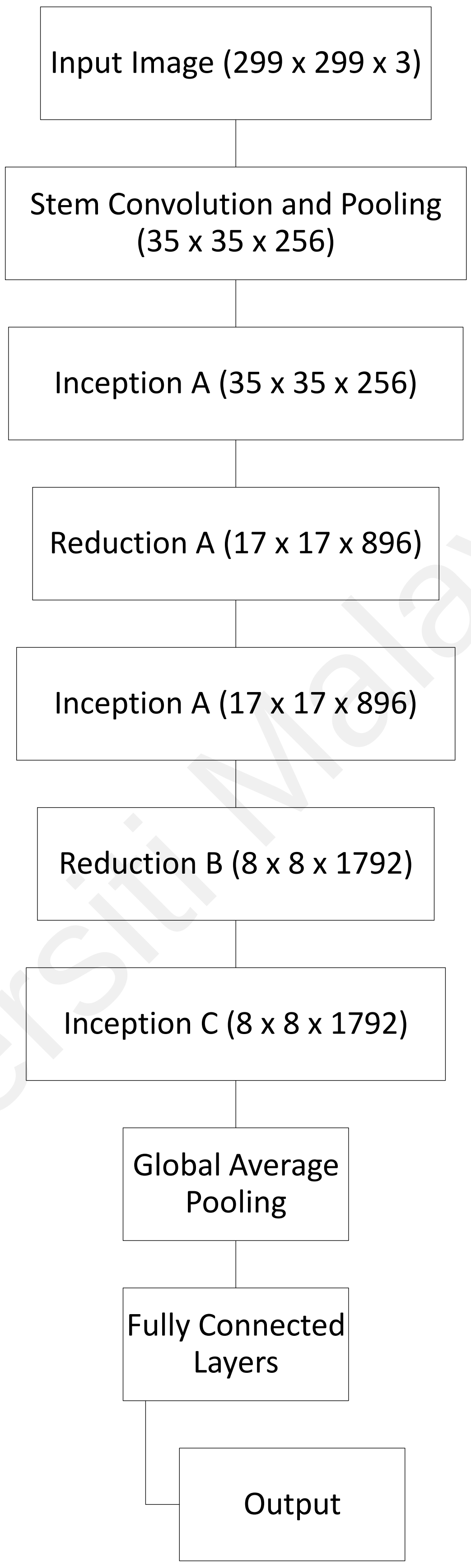

Input Image (299 x 299 x 3)
Stem Convolution and Pooling (35 x 35 x 256)
Inception A (35 x 35 x 256)
Reduction A (17 x 17 x 896)
Inception A (17 x 17 x 896)
Reduction B (8 x 8 x 1792)
Inception C (8 x 8 x 1792)
Global Average Pooling
Fully Connected Layers
Output

Figure 2.8: Inception v-4 Diagram (Szegedy et al., 2016)

Inception v4 architecture is similar to Inception v3 architecture, with several improvements. The upgraded version includes several new design features to improve accuracy, including residual connections, factorised 7 x 7 convolutions, and a stem module that combines convolutional layers and pooling layers to reduce the input size. The residual connections introduced in the ResNet architecture allow the network to learn an identity function and overcome the vanishing gradient problem. The factorised 7 x 7 convolution reduces the computational cost of the network while maintaining accuracy. The stem module aids the network in extracting more valuable features from the input data by decreasing the spatial size.

Figure 2.8 describes Inception v-4. The application of "Inception-A" and "Inception-B" blocks replaces some of the standard inception blocks, includes a more significant number of parallel convolutional layers with varying kernel sizes, and is designed to capture a broader range of features. These reduction blocks use a combination of convolution, max pooling, and average pooling operations to reduce the feature map size while preserving important features.

2.3.1.8 DenseNet

DenseNet, known as the Dense Convolutional Network, aims to improve the channel of data and gradients through the network by introducing dense connections between layers (Huang et al., 2017). In typical neural network architectures, each layer takes input feature maps from all preceding layers, resulting in a dense connectivity pattern. Within the dense layer, each layer is connected to every other layer in a feedforward manner. A dense block typically ends with a transition layer that reduces the number of feature maps and spatial resolution and helps to control the number of parameters in the network. The model has achieved advanced results on various image classifications. Figure 2.9 shows the DenseNet architecture.

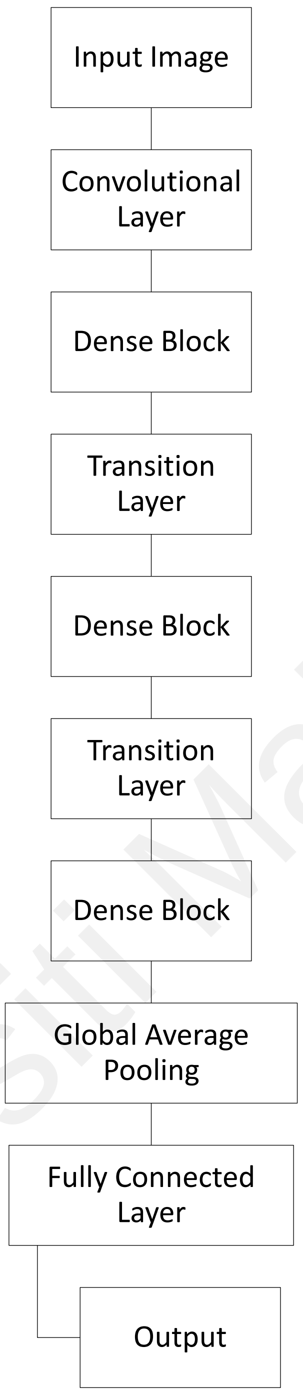


Figure 2.9: DenseNet Diagram (Huang et al., 2017)

2.3.1.9 SENet

The Squeeze-and-Excitation Network is a CNN architecture proposed in 2018 by Chinese researchers. The architecture comprises a novel component called the "Squeeze-and-Excitation block", designed to advance the network's capability to architecture interdependencies between feature channels (Hu et al., 2017). The SENet architecture includes several "SE blocks" within the network. Each "SE block" includes a "squeeze" operation that reduces the spatial dimensions of the feature maps and an "excitation" operation that learns a set of weights to apply to the channel. These weights are learned during training and are used to scale the feature maps to amplify more critical feature maps and suppress less important feature maps (Hu et al., 2017).

The squeeze layer is the global average pooling layer that applies one value for each feature map and condenses the spatial dimension to one value. An MLP that learns a set of weights that scale each feature map according to its significance creates the excitation layer. The following convolutional block in the network receives the scaled feature maps as the final step (Hu et al., 2017).

2.3.1.10 MobileNet

MobileNet is a CNN architecture designed for efficient mobile applications with limited computational resources (Sandler et al., 2019). The model's novelty applies depth-wise separable convolutions that significantly decrease the number of parameters, reducing computational costs (Sandler et al., 2019). Depth-wise convolution applies a single filter of 3 x 3 kernels to each input channel separately. Point-wise convolution is 1 x 1 convolution, also known as network-in-network layer. It changes the number of channels, which refers to the depth of a neural network that assists in transforming the features and decreasing the dimension of the channel space. Depth-wise convolution is often followed by point-wise convolution. (Sandler et al., 2019). It has become a famous architecture for mobile vision applications due to its lightweight architecture to extract features and information within the space and decrease computational cost (Sandler et al., 2019).

MobileNet uses other methods to make it smaller and cheaper to run. For example, it uses a linear bottleneck structure to lower the number of input channels for each depth-wise convolution, a width multiplier to lower the number of output channels for each point-wise convolution, and a global average pooling layer instead of a fully connected layer at the network's end (Sandler et al., 2019).

2.3.1.11 EfficientNet

EfficientNet was presented in 2019 by Google researchers to find the ideal balance between model size, accuracy, and computational efficiency by expanding the network architecture in a principled way (Tan & Le, 2019). The architecture consists of a series of building blocks repeated multiple times, forming a hierarchy of feature extractors. The feature maps' width, depth, and resolution are increased at each level, allowing the model to capture more intricate patterns. At the lowest level, the network consists of a stem convolution layer followed by a series of repeated blocks that include depthwise separable convolutions with fewer parameters than traditional convolutions (Tan & Le, 2019).

The EfficientNet architecture is scaled using a complex scaling method that concurrently expands the network's width, depth, and resolution. Because of the method, RegNet attains higher accuracy with fewer parameters than previous CNN architectures. The basic building block of an EfficientNet network is called an MBConv block, which stands for Mobile Inverted Residual Bottleneck Convolution (Tan & Le, 2019). It consists of a depth-wise separable convolution. EfficientNet uses a series of these blocks, with different numbers of repetitions and different kernel sizes, to create the complete network architecture.

The architecture consists of a series of MBConv blocks with different kernel sizes, expansion factors, output channels, and strides. The network's depth coefficient determines the number of repetitions for each block, while the network's width coefficient determines the output channel size. The network's resolution coefficient determines the resolution of the input image. Additionally, the network's depth, width, and resolution are scaled up or down using a set of scaling coefficients to create a family of EfficientNet models with different sizes and computational costs.

The EfficientNet has a few variants. The number of variants in the name dictates the overall scale of the model. The highest number means that the model is the most accurate of all the other variants.

2.3.1.12 RegNet

RegNet uses systematic design space exploration to optimize the neural network architecture based on simplicity, scalability, and regularization principles (Radosavovic et al., 2020). RegNet consists of a sequence of building blocks called RegNetX and RegNetY, where X and Y represent the number of layers and the width multiplier, respectively. RegNetY networks are wider than RegNetX networks and are designed for larger computation budgets.

The RegNet architecture uses group normalization, which normalizes the activation statistics of a group of features instead of normalizing each feature independently, and residual bottleneck blocks with width-wise convolutions, which reduce the number of input channels before applying a standard convolution. With this strategy, accuracy is maintained while computing complexity is decreased. With fewer parameters and a quicker inference time than other top architectures, the RegNet architecture has exceeded performance expectations on various benchmarks for image classification (Radosavovic et al., 2020).

Convolutional layers with 32 filters and a stride of 2 are applied in the first layer of the RegNet architecture to minimise the input's spatial dimensions, and next is a max pooling layer with a 3 x 3 pool and a stride of 2. Subsequently, the network has a sequence of residual blocks with a progressively more significant number of filters, each consisting of two convolutional layers that skip the residual block. The network uses group normalization after each residual block to normalise the activation statistics of each group of features. Finally, the network uses a global average pooling layer to produce a fixed-size feature vector, followed by a fully connected layer and a softmax activation to produce the classification probabilities for the input image (Radosavovic et al., 2020).

2.3.1.13 ViT Transformer

The ViT transformer method was first applied in natural language processing (NLP), but it can be applied to classify complex images (Dosovitskiy et al., 2020). It is a transformer, which dictates that the model is convolution-free (Chen et al., 2021). It is a transformer-based method with a performance that exceeds CNNs for image classification (Chen et al., 2021). The basic idea of ViT is to represent an image as a sequence of patches, each of which is embedded into a vector using a learnable embedding function. The sequence of patch embeddings is then fed into a transformer encoder, which consists of multiple layers of self-attention and feedforward neural networks. The classification head is usually an FC layer receiving the transformer encoder's output to produce the final prediction.

ViT Transformer divides the input image into non-overlapping ‘p’ patches. Each patch is flattened into a vector and passed through a linear projection layer to get a patch embedding; to obtain a series of token embeddings, the patch embeddings are concatenated and put through a linear projection layer. A classification task can be optimised by adjusting the number of transformer and hidden layers, attention heads, and patch embedding size (Dosovitskiy et al., 2020).

2.3.1.14 ResNeSt

ResNeSt (Residual Neural Network with Split-Attention) is a deep learning architecture inspired by the original ResNet architecture and incorporating a split-attention mechanism to capture the interdependencies between channels in convolutional feature maps effectively (Zhang et al., 2020). The split-attention mechanism involves dividing the channels of a convolutional feature map into multiple groups and more fine-grained relationships between channels, leading to better feature representations and improved performance. ResNeSt also introduces a nested multi-scale architecture, where each network stage is composed of multiple parallel branches that process features at different scales (Zhang et al., 2020), allowing the network to capture information more effectively across a range of spatial scales and resolutions.

The ResBlock is the basic building block of the ResNeSt architecture. It consists of a sequence of operations, including convolution, batch normalisation, and activation, followed by a split-attention mechanism. The ResBlock is repeated multiple times within each stage of the network. The number of ResBlocks in each stage depends on the depth of the network, with deeper networks having more ResBlocks.

2.3.1.15 Swin Transformer

Swin Transformer is a proposed architecture for vision tasks that combines the benefits of the transformer and the convolutional neural network (Liu et al., 2021). Swin Transformer applies a hierarchical architecture with several stages consisting of blocks; each block starts with a layer normalisation operation on the input. Subsequently, a 1 x 1 convolution is applied at the input, dividing the channels into four. The generated feature maps are divided into p x p sized patches on a grid. A convolutional layer of 3 x 3 with filters is implemented (Liu et al., 2021). The resulting feature maps are then aggregated into a single tensor using a 1 x 1 convolution with ‘c’ filters. A connection to bypass the in-between channels is added from the input to the output of the block, which is also normalised. Finally, a ReLU is applied (Liu et al., 2021).

The Swin Transformer architecture is based on a series of these blocks, each with an increasing spatial resolution. The input is downsampled at each stage, increasing the number of channels. The Swin Transformer also uses a shifted window mechanism, which allows overlapping receptive fields and reduces the need for pooling operations. The Swin Transformer has achieved advanced results on several target datasets for image classification and object detection tasks.

2.3.1.16 CrossViT

The CrossViT is a recently proposed computer vision model that combines the Vision Transformer (ViT) architecture with a cross-attention module. The CrossViT architecture comprises a stack of CrossViT blocks, each containing multiple multi-scale attention layers. These layers use a combination of self-attention and cross-attention mechanisms to capture the input image's local and global features. The self-attention layers capture local features, while the cross-attention layers capture global features by attending to features from other spatial locations. The CrossViT block’s multi-scale attention layers produce a collection of attention feature maps from various scales after receiving input feature maps from the input layer before them. The attended feature maps are then processed by a feed-forward network and normalised using layer normalisation. The output of the block is fed into the next block in the stack. Instead of requiring quadratic time, the CrossViT model only requires linear time for computational and memory complexities (Chen et al., 2021). Overall, CrossViT is a powerful and flexible architecture that achieves advanced performance on various image classification benchmarks and is computationally efficient. There are many other variations of ViT transformer models; CrossViT developed a dual-path representation incorporating a vision transformer (Chen et al., 2021).

CrossViT architecture incorporates a multi-scale attention mechanism that operates across multiple levels of feature maps. The architecture especially applies a hierarchical attention mechanism that attends to the feature maps' spatial and channel dimensions at different scales. The multi-scale attention mechanism is composed of two types of attention modules: i) cross-level attention (CLA) and ii) cross-scale attention (CSA). The CLA module attends to the channel dimension across different levels of feature maps, while the CSA module attends to both the spatial and channel dimensions at the same level of feature maps.

The spine of the CrossViT architecture is typically a convolutional neural network that extracts features from the input image. The multi-scale feature maps are obtained by applying convolutional layers with different spatial resolutions and channel dimensions. The multi-scale attention modules then operate on these feature maps to attend to relevant and suppress irrelevant features. Finally, the classification output is produced based on the observed features.

The CLA module takes input feature maps at different levels and computes the channel-wise attention weights for each level. It computes the pairwise similarity between the channels of different levels and uses these similarities to compute the channel-wise attention weights. The attention weights are then applied to the feature maps to obtain the attended feature maps.

The CSA module takes as input the feature maps at the same level and computes the spatial and channel-wise attention weights. The module computes the pairwise similarity between the spatial locations and the channels and uses these similarities to compute the attention weights. The attention weights are then applied to the feature maps to obtain the attended feature maps.

The output feature maps are obtained by combining the attended features from different levels and scales. These attended feature maps are then passed through a set of fully connected layers to produce the classification output.

2.3.1.17 AutoFormer 2021

The AutoFormer is a novel deep-learning architecture for image classification tasks proposed in 2021. The architecture combines convolutional neural networks and transformers using self-attention mechanisms to capture long-range dependencies within images (Wu et al., 2021). The AutoFormer architecture consists of a backbone network of multiple blocks, each containing convolutional and transformer layers. The convolutional layers are used for feature extraction, while the transformer layers are used for modelling the relationships between features. The transformer layers in the AutoFormer architecture use multi-head self-attention to capture the relationships between features at multiple scales. Specifically, the self-attention mechanism is applied to different subsets of feature maps with different spatial resolutions (Wu et al., 2021). This enables the model to capture local and global context within the image. The AutoFormer architecture also includes a novel attention mechanism called “shifted window attention”, which reduces self-attention's computational complexity by limiting the attention range to a local region. The model focuses on relevant features while ignoring irrelevant ones, leading to improved performance.

Each transformer block contains several layers, including multi-head self-attention, convolution, and normalisation layers. The multi-scale attention mechanism is achieved by applying self-attention to different subsets of feature maps with different spatial resolutions. The shifted window attention mechanism is also applied to limit the range of attention. In conclusion, the AutoFormer architecture shows promising results on various image classification benchmarks, demonstrating the effectiveness of combining convolutional and transformer networks.

2.3.2 CNN Architecture Challenges and Limitations

Table 2.2 describes the challenges and limitations of the existing CNN models. ResNet, a convolutional neural network, constructs networks by stacking residual blocks. ResNet has several variations, namely ResNet 50, ResNet 101, and ResNet 152. The performance was proven to be impressive, but it has certain drawbacks and difficulties. The computational density of ResNet 50 is comparable to MobileNets or EfficientNets, which require a disproportionately higher floating point operation per second (FLOPs) (Neural Magic, 2020). The computational density is the reason that ResNet 50 requires GPUs to process. ResNet 50 may suffer from vanishing or exploding gradients that hinder convergence during the training of the neural network (Kaushik, 2022). Accuracy becomes saturated as the depth increases and subsequently declines. Adding more layers to a suitable deep model resulted in a larger training area. The ResNet applied a bottleneck design to minimise the number of parameters and matrix multiplications, so each layer can be trained quickly. Nonetheless, ResNet might reduce the network's capacity for representation (Zuk, 2020). Another CNN architecture that was inspired by ResNet and expanded by adding more dimensions was called ResNeXt. The expansion of the latter model was greater in depth and width, increasing the computational costs even further. Due to the high memory requirement, the ResNeXt model needed significant resources to store intermediate computations and parameters (Masters & Luschi, 2018). The model applied batch normalisation to improve classification performance and convergence speed (Park S. , 2021). However, the number of hyperparameters increases, causing a disparity in the input image's performance between training and inference (Masters & Luschi, 2018). During training, the ResNeXt model applied mini-batch statistics to compute errors and update the model, which significantly affected the stability and convergence of the model (Brownlee, 2019).

Table 2.2: Challenges and Limitations of the Existing Models.

No	Sources	Model Type	Challenges and Limitations
1	(Ardakani et al., 2020; Nguyen et al., 2020; Gozes et al., 2020; He et al., 2015)	ResNet50	• Radiologists had difficulty ascertaining COVID-19 and other atypical and viral pneumonia images because they have similar symptoms. • Deep networks require significant computational resources and memory. • They are overfitting on small datasets or datasets with different characteristics from the training set. • When training intense networks, vanishing or exploding gradient problems might occur, even with skip connections (residual blocks). • The model is prone to overfitting when the dataset is insufficient.
2	(Ardakani et al., 2020; Nguyen et al., 2020; Narin et al., 2021; He et al., 2015)	ResNet101	• Radiologists had difficulty ascertaining COVID-19 and other atypical and viral pneumonia images because they have similar symptoms. • Deeper models suffered from vanishing or exploding gradients. • Small datasets are challenging due to overfitting.
3	(Ardakani et al., 2020; Nguyen et al., 2020; Xie et al., 2017)	ResNeXt101	• Radiologists had difficulty ascertaining COVID-19 and other atypical and viral pneumonia images because they have similar symptoms. • A large number of parameters required substantial computational resources. • The hyperparameter tuning for cardinality is time-consuming.
4	(Ardakani et al., 2020; Nguyen et al., 2020; Huang et al., 2017)	DenseNet	• Radiologists had difficulty ascertaining COVID-19 and other atypical and viral pneumonia images because they have similar symptoms. • A dense connectivity pattern resulted in high memory consumption during training. • Dense connectivity made the model computationally expensive.

Table 2.2: Continued.

5	(Ardakani et al., 2020; Nguyen et al., 2020; Szegedy et al., 2014)	GoogleNet	• Radiologists had difficulty ascertaining COVID-19 and other atypical and viral pneumonia images because they have similar symptoms. • The large number of parameters increases the memory and computational requirements. • The architecture is more complex, causing it to be more challenging during training and fine-tuning. • The auxiliary classifiers during training can introduce additional overhead.
6	(Ardakani et al., 2020; Nguyen et al., 2020; Cheng Jin et al., 2020)	ResNet152	• The challenges are similar to ResNet101 due to the increased depth.
7	(Ardakani et al., 2020; Nguyen et al., 2020; Krizhevsky et al., 2012)	AlexNet	• It is an older architecture compared to more modern models and has limitations in depth and complexity.
8	(Ardakani et al., 2020; Simonyan & Zisserman, 2015)	VGG-16	• The architecture is shallow compared to more modern architecture.
9	(Ardakani et al., 2020; Simonyan & Zisserman, 2015)	VGG-19	• It experienced similar challenges as VGG-16.
10	(Ardakani et al., 2020; Iandola et al., 2016)	SqueezeNet	• The application of aggressive compression techniques reduces the number of parameters, but it might cause a loss in performance.
11	(Ardakani et al., 2020; Sandler et al., 2019)	MobileNet-V2	• The network is designed for efficient mobile and embedded device deployment but may cause lower accuracy than larger models.

Table 2.2: Continued.

12	(Ardakani et al., 2020; Chollet, 2017)	Xception	• The complexity of the model increases the computational cost.

2.4 Deterministic Implementation

The deterministic algorithm in deep learning ensures that the results are reproducible and confirmable. The main goal of the algorithm is to achieve replicability (Nagarajan et al., 2018). The deterministic implementation ensures that the output equals any given input while operating under predetermined experimental conditions (Nagarajan et al., 2018). Because it operates efficiently on machines and is the most used algorithm in DL research. Deterministic implementation is essential because it allows for reproducibility and consistency in the results. Deterministic implementation for CNN can be achieved by setting the seed value for the random number generator to ensure that the same initial conditions are used every time the model runs. However, some operations in the deep learning framework are non-deterministic, which means that each time the algorithm of the neural network runs, the results will always be different, even by providing the same inputs like batch normalisation and dropout. Hence, non-deterministic operations should be avoided.

ResNets, GoogleNet, DenseNet, and ResNeXt models all apply deterministic implementation techniques (He et al., 2015) (Szegedy et al., 2014) (Xie et al., 2017) (Huang et al., 2017). Non-determinism often originates from GPUs as its source. Numerous GPU processes are non-deterministic by nature (Nagarajan et al., 2018). The advantages of deterministic implementation in CNN are reproducibility, consistency, and debugging. Deterministic implementation can reproduce the same output every time the model is generated and run using the same input. With deterministic implementation, the model's performance will be consistent across different machines and environments, making it easier to debug and troubleshoot the model. However, the disadvantages of deterministic implementation are computational cost, lack of robustness, stability issues, and lack of diversity.

2.5 Performance Evaluation

The prediction algorithm is assessed during classification to gauge the effectiveness of the classification model. The performance evaluation determines if the training model and classifier can handle test data. Predictive accuracy is typically used as a baseline for evaluation, but the model's categorisation credibility cannot be solely determined by accuracy.

A confusion matrix table was created to measure the classification model's effectiveness, known as a classifier (Kevin Markham, 2014). The table can be divided into four categories for binary classifiers, i.e., true positive (TP), false positive (FP), true negative (TN), and false negative (FN), arranged in a 2 x 2 table. TP shows that the model was predicted and matched with the 'normal' label. TN was correctly predicted and matched with the 'COVID-19 pneumonia' label. The confusion matrix table shows the following evaluation indicators: accuracy, misclassification rate/error rate, true-positive rate/sensitivity/recall, false-positive rate, true-negative rate/specificity, precision, prevalence, and F1 Score. The evaluation metrics were computed in detail since accuracy alone could not determine the classifier's performance.

Performance evaluation assessed the model's ability to classify instances across several classes for multi-class classification. TP happens when the model predicts the correct target class, and TN happens when the model correctly predicts it as not the targeted class.

2.6 Summary

The meta-analysis discovered that most COVID-19 CT features consist of GGO and consolidation at the bilateral lungs in a peripheral distribution and/or septal thickening to create a “crazy-paving” pattern (Bao et al., 2020; Simpson et al., 2020; Duzgun et al., 2020). A multitude of distinct lung disorders may appear as longitudinal changes in conventional CT findings and uncommon findings such as air bronchograms, CT halo signals, and reverse halo signs. (Duzgun et al., 2020). The CT imaging features of COVID-19 pneumonia and their variations in different stages and severity will affect the morphology and, consequently, the quality of interpretation. Other viral pneumonias are found to have bilateral perihilar peribronchial thickening and interstitials that trap the air within the lungs (Weerakkody, 2022). Several nonviral infectious and inflammatory conditions share similarities with the imaging findings of viral pneumonia (Koo et al., 2018). The aetiology of pneumonia is similar among viruses belonging to the same viral family, and there are clear differences in the imaging patterns (Koo et al., 2018).

Several robust CNN models were developed using a voting combinations-based ensemble of fine-tuned CNN models, like VGGNet, ResNet, GoogLeNet, and Inception (Tasci, 2020) to classify chest CT images as positive or negative for COVID-19. Some methods were introduced to detect probable chest anomalies, but a few have addressed the usage of lightweight CNN models for the segmentation of chest CT images (Iyer et al., 2021). This research provides insights and the potential for the classification of COVID-19 chest CT images, and these architectures are still under development and further evaluation before they can be widely applied in public.

CHAPTER 3: METHODOLOGY

3.1 Introduction

The research develops a deep learning-based classification model to automate the binary and multi-class classification of ‘COVID-19 pneumonia’ vs. ‘normal’ and ‘pneumonia’ CT images. There are three stages in executing the methodology. The first stage is a feasibility study using various DL algorithms to classify CT images. The second stage to train existing ImageNet models and a new model. The stage includes pre-processing the CT image datasets to ensure the data are standardised, performing a transfer learning strategy to train the ImageNet models, and training the new model from scratch using the pre-processed dataset. The third stage is to evaluate the performance of all the trained models.

3.2 Research Approach

The method is based on supervised deep learning algorithms to conduct a multiclass classification of CT images of ‘COVID-19 pneumonia’ vs. ‘normal’ and ‘pneumonia’ CT images. Figure 3.1 shows the research workflow. The methodology was categorised into four phases: i) data preparation; ii) data augmentation; iii) design of the proposed model; and iv) result evaluation.

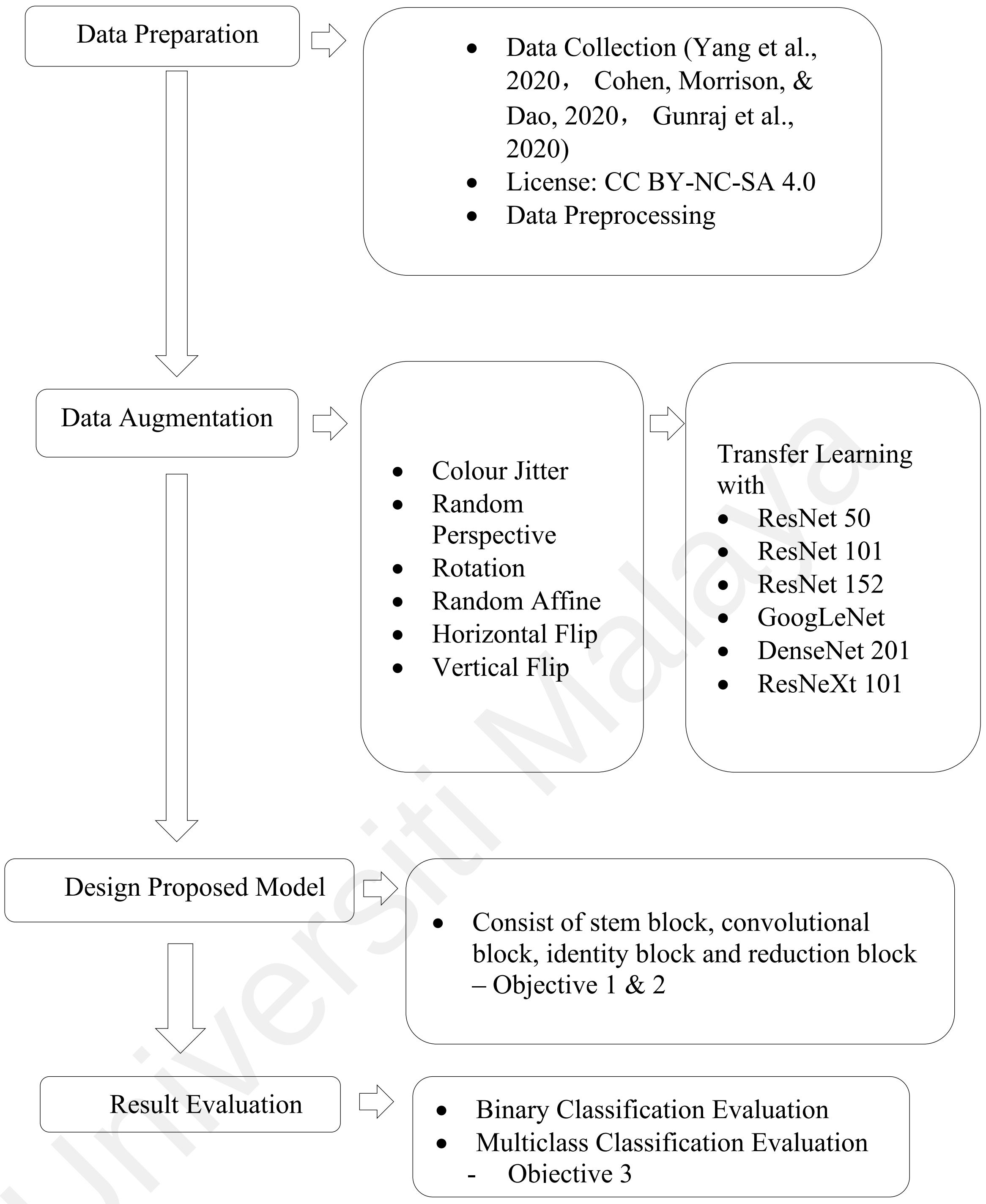


Figure 3.1: Research Workflow

3.2.1 Materials and Tools

The research used the PyTorch framework running in Jupyter Notebook. In addition, Google Colab and Kaggle were utilised for image training, validation, and testing. Google Colab is a cloud-based development environment provided by Google that uses a Jupyter Notebook interface. On the other hand, Kaggle is a popular online community platform for data science enthusiasts, machine learning practitioners, and AI researchers. It is an online platform that hosts data science competitions, provides datasets and tools for data analysis, and allows users to share and collaborate on data science projects. Table 3.1 describes the summary of both the Google Colab and Kaggle hardware specifications.

Table 3.1: Kaggle and Google Colab Hardware Specification (Kazemnejad, 2019).

No	Parameter	Kaggle	Google Colab Pro
1	GPU	Nvidia P100	T4
2	GPU Memory	16 GB	16GB
3	GPU Memory Clock	1.32 GHz	1.59 GHz
4	Performance	9.3 TFLOPS	8.1 TFLOPS
5	Support Mixed Precision	No	Yes
6	GPU Release Year	2016	2018
7	Number of CPU Cores	2	2
8	Available RAM	12 GB	26.75 GB
9	Disk Space	20 GB	120 GB

3.2.2 Hyperparameters

The hyperparameters were trained to attain accuracy once the neural networks had been defined. The learning rate was set at 0.1, and a lengthy training period was necessary. The momentum was fixed at 0.9 to avoid oscillations. The settings controlled the training speed while allowing the network to adapt and recognise the next step from the prior training. The number of sub-samples provided to the network in each epoch is called the batch size. The training is terminated if the validation loss values show no improvement after 10 iterations. DL hyperparameter configurations are crucial to achieving optimal results. Therefore, a few assays were conducted to determine the best epoch number. The models were configured to run for 100 epochs at the beginning. However, the difference between the training and validation losses revealed that the models were underfitting and still had the potential to improve. Hence, for binary classification, 300 epochs were executed when the training and validation losses in the dataset acquired for binary classification were minimal. For multi-class classification, 100 epochs were executed to train and validate the models, as the dataset is larger than the binary classification process.

The SGD optimiser minimises the loss function by training the network while updating the computed gradient parameter. Backpropagation calculated the gradients during training, and the optimizer learned using those gradients (Kevin Markham, 2014). The learning rate would decrease by 0.1 every nine epochs because of the weight decay specified at 0.1. The projected probability diverging from the actual label increased due to the cross-entropy loss, which decreased the loss and enhanced training.

3.3 Data Acquisition

The radiological imaging data source described in this section is for researchers to use as input data for CNN architectures. LeNet, AlexNet, GoogleNet, VGGNet, and ResNet are several CNN-based models that can be used to process high-dimensional radiology images like CT images (Nguyen et al., 2020). Table 3.2 describes the available data sources to use.

Table 3.2: Data Sources on COVID-19 CT images.

No.	Sources/Links	License Code	Data Type	No. of Images
1	(Gianluca Maguolo, 2020) (Tartaglione et al., 2020) https://www.kaggle.com/bachrr/covid-chest-xray	CC0 1.0 Public Domain	Posterior-anterior (PA), Anterior-posterior (AP) lateral for X-rays & Axial or Coronal for CT scans	Normal images: 1,576, Pneumonia ARDS images: 2, Viral Pneumonia images: 1,493, COVID-19 images: 58, SARS images: 4, Bacterial Pneumonia images: 2,772, Bacterial Streptococcus images: 5
2	(Eurorad, 2020) https://www.eurorad.org/advanced-search?search=COVID	CC BY-NC-SA 4.0	COVID-19 CT images of the lungs for infants and children	COVID-19 images: 49
3	(Yang et al., 2020) https://github.com/UCSD-AI4H/COVID-CT	Unknown	349 CT images containing clinical findings of COVID-19 from 216 patients,	COVID-19 images: 349, Non-COVID-19 images: 397
4	(Gunraj et al., 2020) https://github.com/haydengunraj/COVIDNet-CT	CNCB: CC BY 3.0 CN DEED Attribution 3.0 China Mainland CT Images in COVID-19 (TCIA): CC BY 4.0 DEED Attribution 4.0 International COVID-19 CT Lung and Infection Segmentation Dataset: CC BY-NC-SA 4.0 DEED Attribution-NonCommercial-ShareAlike 4.0 International Data	COVIDx CT-3A and CT-3B datasets, comprising 425,024 CT slices from 5,312 patients and 431,205 CT slices from 6,068 patients	COVIDx CT-3A: Normal CT images: 71,488, Pneumonia CT images: 42,943, COVID-19 images: 310,593 COVIDx CT-3B: Normal CT images: 71,488, Pneumonia CT images: 42,943, COVID-19 images: 316,774

		from The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): CC BY 3.0 DEED Attribution 3.0 Unported		
5	(Jenssen, 2020) http://medicalsegmentation.com/covid19/	Unknown	100 axial CT images from more than 40 patients with COVID-19	COVID-19 images: 100

Data acquisition was the first step in developing and creating the algorithm for COVID-19 detection. There are existing large datasets for the public to utilise. The biggest obstacle to training and validating the proposed model is the availability of authentic, annotated, and class-balanced CT images. Most available open-sourced datasets are unorganised and highly biased towards a particular class. Moreover, the authenticity of the open-sourced website is often questionable. However, there was no collection of COVID-19 chest X-rays or CT scans designed for computational analysis until it was assembled from websites and publications. (Cohen et al., 2020). Due to privacy reasons, CT images are challenging to acquire, which is a tremendous obstacle towards AI-powered research and development. Initially, the data were acquired from GitHub (Yang, et al., 2020). The usage of the dataset was confirmed by a senior radiologist in Tongji Hospital, Wuhan, China, since the outbreak of the COVID-19 pandemic in China (Yang et al., 2020). Hence, it was suitable to conduct experimental studies that further demonstrate the CT images' usefulness. However, the availability of the COVID-19 CT scan images was minimal and insufficient. Besides, based on the senior radiologist, it was stated that CT images could be utilised to determine if the patient is infected with viral pneumonia. However, CT images cannot determine which virus is causing the pneumonia infection.

Moreover, the CT images available on GitHub, the open-source platform, raise concerns about the usability of the dataset because the quality of the images is severely degraded, which may result in the diagnostic decisions being less accurate. The Hounsfield Unit (HU) values are lost, and the number of bits per pixel is reduced, causing the overall image's resolution to be reduced. On top of that, only a few critical slices of the original CT images were selected to be made available, which may also have negative implications for the diagnostics. Table 3.3 shows the total number of CT images available for computer analysis is 746, with 349 images labelled ‘COVID-19 pneumonia’ and 397 CT images labelled ‘normal’. Training CNN models on such a small number of images will overfit the deep-learning algorithms. Hence, data augmentation was implemented to create more CT images by varying the existing limited images using techniques such as image rotation, sheering, panning, etc. More details regarding this are presented in the Data Pre-processing section.

Table 3.3: The GitHub Dataset.

Classes	**Datasets**	**Training Set**	**Validation Set**	**Test Set**
COVID-19 pneumonia	349	280	35	34
Normal	397	318	40	39

The second dataset was collected from the COVID-Net Open Source Initiative (COVIDxCT -3), updated on June 2, 2022. Each CT image was associated with captions to differentiate between ‘COVID-CT pneumonia’, ‘normal’, and ‘pneumonia’. The data were available on Kaggle, an online community platform for data scientists and machine learning practitioners to explore, compete, and collaborate on data science projects.

The total number of COVID-19 pneumonia CT images is 425,024 slices from 5,312 patients. The dataset was derived from CT imaging data collected by the China National Center for Bioinformation (CNCB) and cleaned with additional annotations to allow COVID-19 detection methods to be compared across standard datasets. It comprised CT images from different hospitals across China and consisted of three different infection types: 'COVID-19 pneumonia', 'pneumonia', and 'normal' CT images that acted as a control for the research. The CT images were selected with subtle visual differences between 'COVID-19 pneumonia' and 'pneumonia' as they were marked as containing lung abnormalities. The CT images were standardised using an automatic cropping algorithm to crop the image. The background of the CT images had been removed to depict the segmented lung regions, as the contrast present in these images can lead to biases in the model during training, validation, and testing.

The CT image datasets are not novel, and the authors that compiled the images together have cleaned the data and provided additional annotations to allow other researchers to compare COVID-19 deep learning detection models using standard sets of images (Gunraj et al., 2020). The CNCB data contained chest CT examinations from various hospital cohorts across China, consisting of three infection categories. Table 3.4 shows the distribution of the CT images in classes, training, validation, and test sets using the 8:1:1 ratio.

Table 3.4: The Partial Kaggle Dataset.

Classes	Datasets	Training Set	Validation Set	Test Set
Normal	1,682	1,346	168	168
Pneumonia	1,011	809	101	101
COVID-19 Pneumonia	7,308	5,846	731	731
Total	10,001	8,001	1000	1000

Note that this proposed research only uses a portion of the whole dataset. The main reason is the hardware limitations of Google Colab and Kaggle. This limitations caused the change from using Google Colab to using Kaggle to train and test all the models using this new dataset. The following section in Data Pre-Processing elaborates on the greater length of this limitation. In summary, this small portion of the dataset already far exceeds the number of CT images in the previous dataset (10,001 CT images versus 746 CT images).

3.4 Data Pre-Processing

Data pre-processing is an essential step in deep learning that involves transforming unstructured data into a format that a neural network can effectively use. Data cleaning involves handling missing values, outliers, and noise in the input data. The data were collected from various studies; the images' dimensions, contrast, and intensity needed consistency. Some images were unsuitable for pre-processing as the lung density appeared utterly black. Hence, smoothing and filtering the input data at the beginning is essential.

3.4.1 Binary Classification

Firstly, data augmentation is performed on the GitHub dataset, with training, validation, and test CT images split into their own folder. Data augmentation is proper when working with limited training data like the one from GitHub. It can increase the quantity of the training samples using rotations, horizontal flips, vertical flips, translation, colour jitters, random perspective, random affine, and auto-augmentation in the x and y-axis (PyTorch, 2017). The colour jitter arbitrarily alters the images' brightness, saturation, hues, contrasts, and other properties. The random perspective transform function allows the image’s perspective to be randomly distorted and scaled at different angles. The rotation transform function rotates the images within the range of 30° to 70°.

It is important to note that the ratio between COVID-19 and non-COVID-19 images is maintained before and after augmentation (stratified augmentation). On top of that, care is taken so that these transformations maintain the quality of the image and do not hinder the radiologist’s ability to interpret. Table 3.5 and Figure 3.2 summarize all the data augmentation information explained.

Table 3.5: The Number of GitHub Datasets after Data Pre-processing and Augmentation.

Classes	**Datasets**	**Training Set**	**Validation Set**	**Test Set**
COVID-19 (1)	2,094	1,680	210	204
Non-COVID-19 (0)	2,382	1,908	240	234
Total	4,476	3,588	450	438

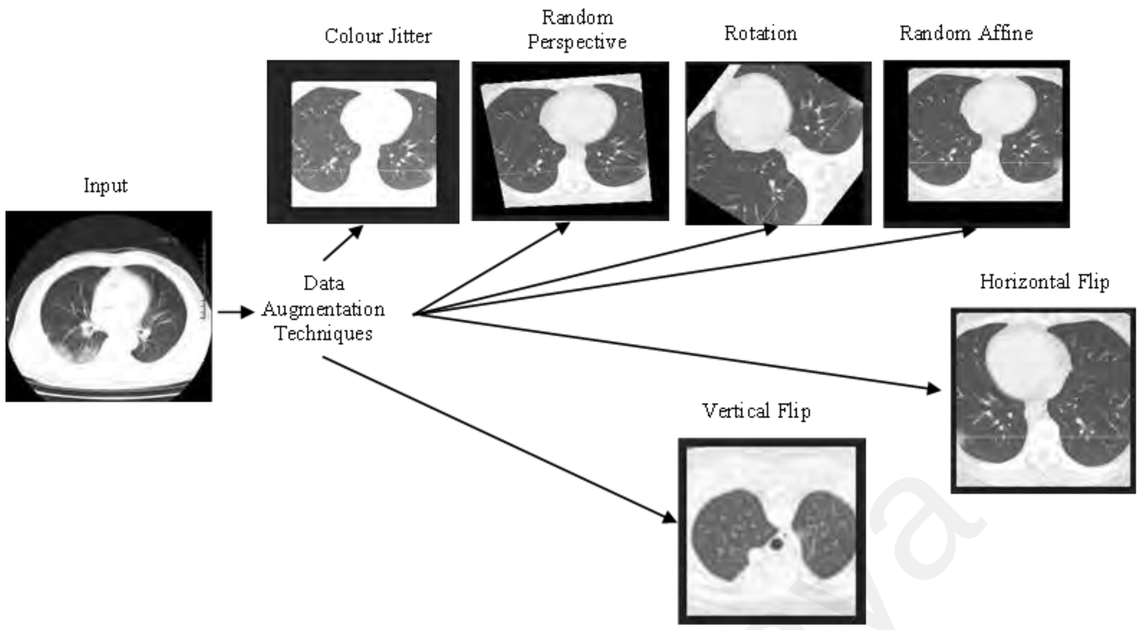


Figure 3.2: Data Augmentation Techniques

After data augmentation, standardisation is the final step in the data pre-processing of this binary classification dataset. Initially, the CT images were encoded into binary classes of '0' and '1' for non-COVID-19 and COVID-19 images, respectively. Next is to standardise the images after encoding the classes from string to integer. The integer representation is used instead of a string because of its ease of use in calculations and because it uses less memory.

Firstly, the images had different dimensions and needed to be resized to 224 x 224 pixels. This resolution was chosen because it is the original resolution used by most ImageNet models. During image resizing, the ratio aspects of the image remained to avoid compromising the details of the images. These images are first resized to 256 x 256 pixels before being centre-cropped to 224 x 224 pixels. The resize is done in two steps to "zoom in" to the lung instead of having many empty spaces with a simple single-step resize.

Finally, the images were normalised using minimum-maximum normalisation techniques with a mean and standard deviation. The normalisation method eliminates the bias from the features and datasets within the range. To further elaborate, normalisation can accelerate the training process and improve convergence to ensure that larger magnitudes do not dominate the learning process. The technique of sample-wise normalisation or instance-wise normalisation was applied, where each sample was scaled independently. The objective of this method was to ensure that each sample has a similar scale and range, which aids the convergence performance of deep learning models. Normalising the input images will prevent any bias arising from variations in the magnitude or range of different samples. Mathematically, the sample-wise normalisation for a single sample x can be represented as:

$$\boldsymbol{x_{normalised}} = \frac{\boldsymbol{x-\mu}}{\boldsymbol{\sigma}} \quad \textbf{3.1}$$

Where $x_{normalised}$ is the normalised version of sample x, μ is the mean of the sample's values, and σ is the standard deviation of the sample's value.

Resizing, centre-cropping, and normalizing are implemented as PyTorch Transformers for ease of usage with the PyTorch data loader. The sample-wise normalisation applies independently to each dataset sample; the mean and standard deviation used for normalisation are calculated based on the values within the same sample, not across different samples. This technique was applied because the image data acquired exhibited significant variations in magnitude and scale. The normalisation method aided in the removal of sample-specific biases and ensured that the model treated each sample equally during training.

3.4.2 Multi-class Classification

Data augmentation is unnecessary for the Kaggle datasets as the number of CT images is more than sufficient. However, a custom PyTorch dataset must be implemented because the number of images is too numerous to manually split into the training, validation, and testing datasets. The custom dataset will only need to be instantiated once and will stay in the memory for usage through the Kaggle Jupyter Notebook session.

First, the custom dataset needs to override PyTorch Dataset functions such as "_getitem_" and "_len_". Then, additional functions need to be implemented to handle reading the metadata of the Kaggle dataset to identify which classes an image belongs to. Much like the GitHub dataset, the Kaggle dataset classes are also represented by integers. They are also encoded into the multiclass categorisation of '0', 1, and 2 for standard lung CT images, pneumonia, and COVID-19. Finally, a function to perform stratified removal of images from the dataset is implemented. The purpose of this removal function is to provide the ability to set the maximum number of images this custom dataset will process.

Once the custom dataset is implemented, it reads the Kaggle dataset by passing in the dataset folder path, the file that contains the metadata, and the maximum number of images this custom dataset will produce. In this research, the dataset is limited to a maximum of 10,000 images due to the hardware limitations of the Kaggle virtual machine. In this step, Kaggle was applied instead of Google Colab. The benefit is that the Kaggle dataset is available automatically in the Kaggle Jupyter Notebook without explicitly downloading it. The only requirement is that the dataset has to be hosted on Kaggle.

Once the custom dataset reads the entire dataset, the "train_test_split" function from the 'sklearn' Python package splits the dataset into train, validation, and test. As stated in the previous section, 3.3 Data Acquisition, the split is performed in the ratio of "8: 1: 1" for "train: validation: test", respectively. Note that, despite this split being random by default, the same seed used throughout this research is passed into the "train_test_split" function to ensure reproducibility.

Finally, a wrapper dataset is implemented to wrap over the custom dataset. This wrapper dataset provides the ability to specify what transformations to perform on the images and, optionally, even the class label. Another dedicated wrapper dataset is also written specifically to provide the ability to provide the file path to each of the images.

The transformations are applied to the CT images using a wrapper dataset, i.e., convert the images to grayscale, resize to 512 x 512 pixels, and normalise. Note that despite the image being resized to 512 x 512 pixels, the final image size is determined by the bounding box size specified in the metadata of the Kaggle dataset. For the training dataset, additional image transformations are applied: random horizontal flip, random verticle flip, random rotation, which is limited to at most 10 degrees, colour jitter, random perspective, and random affine. These transformations are applied only to the training dataset to prevent the model from overfitting the training images.

3.5 Proposed Model Architecture Design

The critical consideration for neural networks is to determine the architecture of the overall network. The number of layers and how the layers are connected and arranged to form a structure are considered during the design of the proposed model.

This research highlights the proposed neural network architecture and the hyperparameters. In this study, the ResNet and Inception models were the baseline for the proposed model. The proposed model was inspired by skipped connections and consisted of deep layers of CNN. The proposed network architecture is shown in Figure 3.3.

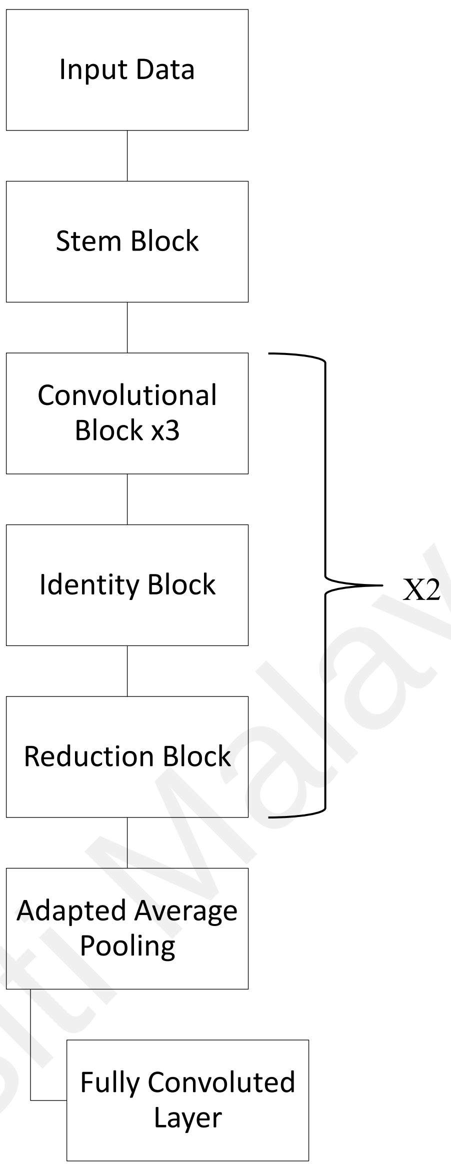


Figure 3.3: Proposed Model Design

The proposed model consists of modules combining a stem block, followed by repeated layers of convolutional, identity, and reduction blocks. The magnitude of the sets transform is its priority, rather than increasing the depth and width of the module.

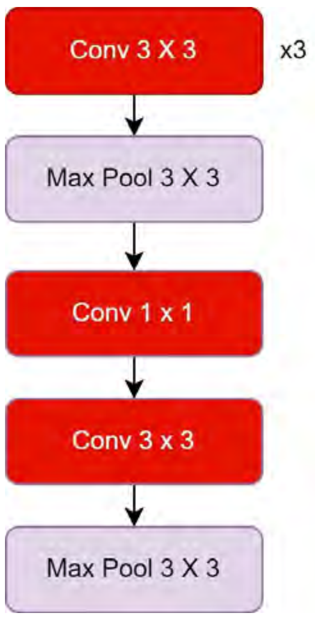


Figure 3.4: Stem Block Design

Figure 3.4 describes the design of the stem block in the proposed model. It is the entry block of the proposed model at the beginning of the CNN architecture that initiates the flow of information through the network. The stem block performs initial pre-processing and feature extraction on the input data before passing through the next layer. It consists of a sequence of convolutional, pooling, and normalisation layers. The max pooling layers within the stem block reduce the spatial dimensions of the input to down-sample the feature maps.

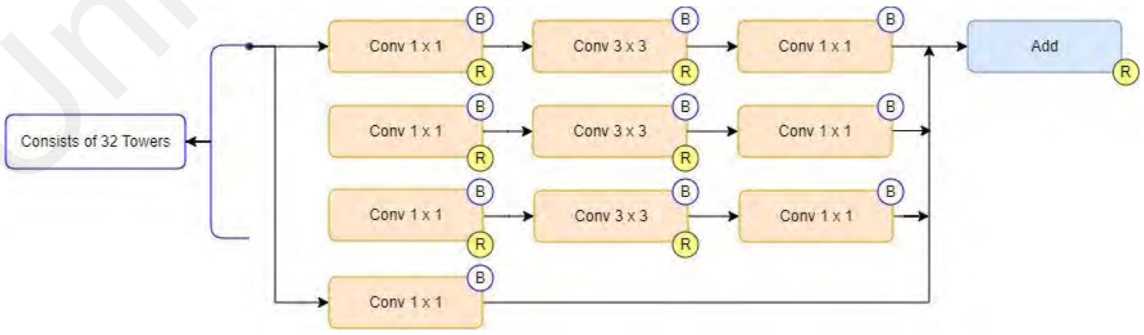


Figure 3.5: Convolutional Block Design

Figure 3.5 describes the design of the convolution block. Each convolution sequence within the convolution block applied batch normalisation, leaky ReLU, and max pooling.

The proposed model technique configures several convolution sequences in parallel, effectively making a broader network to distribute the computational load across multiple processing units to increase computational efficiency. The capacity increase reduces overfitting in the neural network and enhances the network's learning capacity. The wide neural network design allows the network to explore multiple perspectives and capture various feature aspects of the input data, creating more robust and generalised representations.

To elaborate more on Figure 3.5, 32 channels are parallel against each other with the same structure using a 1 x 1 filter convolutional block, a 3 x 3 filter convolutional block, and another 1 x 1 filter convolutional block, also known as the filter-expansion layer, without activation. The filter expansion layer is needed to scale the filter's dimension to match the input's depth. Each channel can focus on capturing specific aspects or modalities of the data, allowing for more effective integration of information and improved representational learning. Introducing redundancies within the neural network will make it less likely to become overly specialised to the training data, resulting in better generalisation performance. The last step involves adding these 32 parallel paths, giving us a single output, before adding the initial input to create a residual connection.

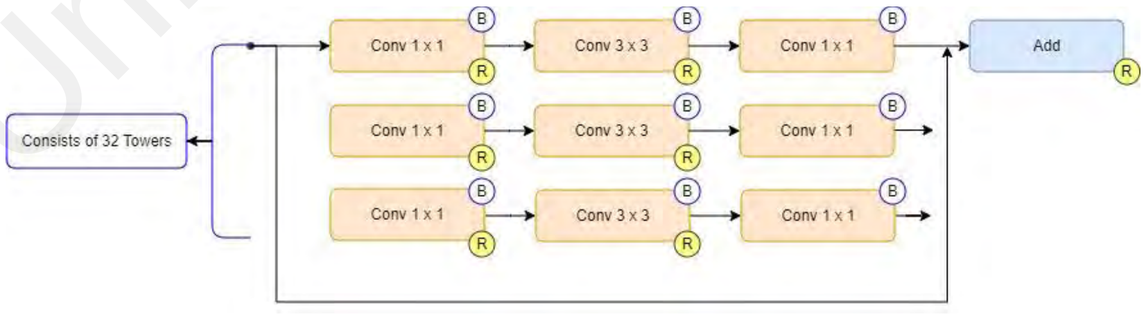


Figure 3.6: Identity Block Design

Figure 3.6 describes the design of the identity block. The identity, also known as the residual blocks, addresses the vanishing gradient problem and effectively trains deep neural networks. The convolutional layer with a 1 x 1 filter adjusts the number of channels and transforms the input presentation. The 3 x 3 filter convolutional layer captures intricate spatial patterns and features of the input data, and the final convolutional layer with a 1 x 1 filter size adjusts the number of channels in the feature maps to match the desired output. The activation function, such as Leaky ReLU, was applied to learn complex relationships. The skip connection in the identity block provided a residual connection, bypassing the convolutional layers and retaining the original information. The skip connection alleviates the problem of vanishing gradients and facilitates the learning process.

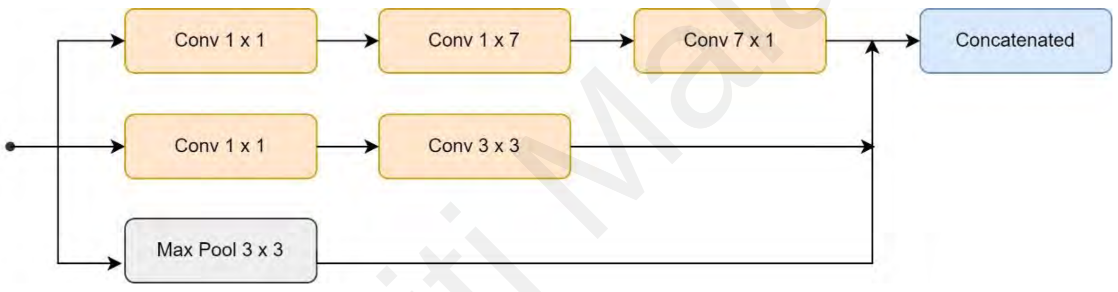


Figure 3.7: Reduction Block Design

Figure 3.7 describes the design of the reduction block. The purpose of this block is to reduce the spatial dimensions of the input feature maps while increasing the number of channels. It down-samples the feature maps and captures more abstract and higher-level features. The reduction block uses pooling operations to downscale spatial dimensions and uses convolutional layers to increase the number of channels. This downscaling process condenses spatial information while preserving essential features. The convolutional filter 1 x 1 is a pointwise convolution used to adjust the number of channels in the feature maps while maintaining the spatial dimension. It performs a linear transformation on the input data by computing the weighted sums of the input channels. When the number of channels is altered, the model can learn different feature combinations and capture higher-level representations. The convolutional filters 1 x 7 and

7 x 1 are depth-wise separable convos that reduce computational costs while capturing spatial and channel dimension information.

3.6 Training and Validation

During the training and validation process, the main architectural models were utilised based on the depth of the network and the width of each layer. A network with a hidden layer can be used for training the input dataset. The study utilises existing architecture models, which are deep neural networks. It used fewer units per layer and fewer parameters and generalised the classification when categorising the test set. Training and validation processes are conducted with constant trial and error to monitor the performance of training and validation set errors before officially running the test (Goodfellow et al., 2016). ResNet50, ResNet101, ResNet152, ResNeXt101, GoogleNet, and DenseNet201 were chosen for transfer learning based on their deep architectures and statistical reasons. We chose this deep learning model algorithm explicitly, stating our prior beliefs about what function the algorithm should learn. Hypothetically, the greater the depth, the better the result in generalising a comprehensive variety of tasks (Goodfellow et al., 2016; Bengio & LeCun, 2007; Bengio & Delalleau, 2009; Erhan et al., 2009; Cireşan et al., 2012; Krizhevsky et al., 2012; Sermanet et al., 2012; Farabet et al., 2013); (Couprie et al., 2013; Kahou et al., 2013; Goodfellow et al., 2013; Szegedy et al., 2016).

3.6.1 Transfer Learning in Binary Classification

Transfer learning for CNN refers to using a trained CNN in a dataset where the number of classes to be identified differs from the initial dataset because it has been applied to various tasks with varied datasets. The technique enhances the targeted domain by utilising knowledge from the source domain and learning challenges; this strategy employs trained model knowledge to learn another dataset (Koval et al., 2023) (Marques et al., 2022).

After processing the data and establishing the design of the proposed model, the next steps are training and validation. The binary classification dataset was acquired from GitHub (Yang et al., 2020). The dataset is publicly available and was collected to conduct deep learning classification. A senior radiologist at Tongji Hospital in Wuhan, China, who handled the medical treatment and diagnosis of many COVID-19 patients between January and April during the pandemic of this disease, has confirmed the applicability of the dataset (Yang et al., 2020).

The dataset collection contains 397 CT images of non-COVID-19 and 349 CT images with clinical findings of COVID-19 from 216 patients (Yang et al., 2020). Since the cohort of patients was from different provinces of China, the characteristics of COVID-19 infection observed in the chest CT images could not represent patients worldwide (Gunraj et al., 2020). Hence, the dataset size is tiny, and data augmentations were conducted to increase the dataset and introduce variations. The augmentation variants included rotations, horizontal and vertical flips, translations, colour jitters, random perspective, random affine, and auto-augmentative in the x and y-axis. The distribution of classes is balanced in terms of quantity. The dataset was separated using an 8:1:1 ratio for training, validation, and testing. The images were assigned to training, validation, and test folders for COVID-19 and non-COVID-19, respectively. The 'torch' data loader automatically categorises the images into classes based on the assigned folders. The images were resized to standardise the dimensions of the images. It was attached to a square template utilising Python's "Pillow (PIL)" image processing package.

During normalisation transformation, the image data were normalised with mean and standard deviation, with [0.485, 0.456, 0.406] representing the mean values of the three colour channels (red, blue, and green) and [0.229, 0.224, 0.225] representing the standard deviation values for each colour channel. The image input model will be 224 x 224 x 3. The dataset has been manually segmented and labelled on GitHub, so annotation and labelling were unnecessary (Yang et al., 2020). The normalisation technique eliminates the bias from the features and the datasets inside the range.

The 'COVID-19 pneumonia' and 'normal' pulmonary CT images were trained using the existing CNN models with pre-trained weights from ImageNet and then later fine-tuned to suit the input data to achieve ideal results. The hyperparameters used for training the binary images using the existing models were the learning rate = 0.0001, momentum = 0.9, number of epochs = 300, batch size = 64, step size = 7, optimizer = stochastic gradient descent, and loss function = cross-entropy loss.

3.6.2 Transfer Learning in Multi-class Classification

After processing the data and establishing the design of the proposed model, the next steps are training and validation. The multi-class classification dataset was acquired from Kaggle (Gunraj et al., 2020). The dataset is publicly available and was collected to conduct deep learning classification. The original dataset consists of chest CT images collected by the China National Center for Bioinformation (CNCB) (Zhang et al., 2020). The COVID-19 CT-3A has many CT scans. The patient cohort increased in quantity and diversity because the original dataset was collected in different provinces of China. Hence, several patient cohorts worldwide were collected to achieve diverse, well-rounded neural network training (Gunraj et al., 2020).

The COVIDx CT-3A dataset used stratified split, a common technique applied in machine learning and data analysis, mainly when dealing with imbalanced datasets or classification tasks. With stratified splitting, the dataset is divided into training, validation, and test sets while maintaining the proportion of classes in each subset similar to the original dataset. The objective of the stratified split is to ensure that the class distribution remains constant between the training, validation, and test sets. 10,001 CT images of the 425,024 images were selected for training, validation and testing. However, even with the increased quantity of data, the ratio between the data classes was imbalanced. Hence, the stratified split method can mitigate biased model performance evaluations, even if we cannot represent the overall class distribution. The method ensures a fair evaluation of the model's performance on all classes and helps prevent overfitting to the majority class in imbalanced datasets. Data augmentation was applied not to create additional samples but to introduce variations in the data for the model to become more robust and generalise unseen data to prevent overfitting. Data augmentation can adequately improve the model's performance; it provides additional variations of the same class, allowing better class discrimination.

During normalisation transformation, the image data were normalised with a mean and standard deviation of [0.5, 0.5, 0.5] representing the mean values of the three colour channels (red, blue, and green) and [0.5, 0.5, 0.5] representing the standard deviation values for each colour channel. The value of 0.5 for both mean and standard deviation ensured that the input data had a consistent and appropriate scale and distribution to stabilise the training process and improve convergence when training the model. The transformation shifts the pixel values to be centred within the range [-1, 1]. The transformation process also converted the colour images into grayscale images. The number of outputs was specified to have three channels where all three channels have the same value, effectively creating a grayscale version of the RGB image. The image input model was resized to 512 x 512 x 3. However, a bounding box zoomed into the relevant area of the CT images.

The CT images will be trained using the existing CNN models. The CT images were trained using existing models and subsequently trained using the proposed model. The hyperparameters used for training the images using the existing models are: learning rate = 0.1, momentum = 0.9, number of epochs = 100, batch size = 16, step size = 7, optimizer = stochastic gradient descent, and loss function = cross-entropy loss. Training and validation also utilised Automatic Mixed Precision (AMP) to accelerate model training and reduce memory consumption by leveraging a mix of low-precision numerical representations for various computation parts. The general idea of applying AMP is to use lower precision, where it is less likely to cause a significant loss of accuracy, while still utilizing higher precision, where it is necessary to maintain accuracy.

3.6.3 Existing Models and Proposed CNN Models

The study consists of a transfer learning strategy with deep learning CNN models, which requires significant data to prevent overfitting in a complex network. The research included deterministic implementation, where the pre-trained weights of the neural networks are fixed during the fine-tuning or transfer learning process. Throughout the process, the weights of the pre-trained model are not updated or modified based on the new dataset used as the input. The deterministic approach applied in transfer learning implemented the pre-trained model and fixed its feature extractor. The final layers are added or modified to the new tasks or dataset applied to the model. The pre-trained weights are learned representations that capture different CT images' standard features and patterns. Using fixed pre-trained weights can avoid overfitting, especially when the new dataset is small and insufficient for training from scratch.

The existing CNN models, like DenseNet 201 and ResNet 101, occasionally have to explode gradients, where the gradients during the backpropagation process become extremely large. It resulted in unstable training and difficulties in converging to an optimal solution, destabilising the learning process. The method we applied to solve this issue was gradient clipping. Gradient clipping is commonly used to clip the gradients to a predefined maximum value. A threshold was set and scaled down the gradient if the norm exceeded it. The method prevents the gradients from becoming too large and stabilises the training process.

The proposed model was designed to overcome existing models' challenges and limitations. It trained the CT images from scratch. Unlike the existing models, the weights and hyperparameters of the proposed models were tuned accordingly to achieve optimal results.

CHAPTER 4: RESULTS AND DISCUSSIONS

4.1 Results and Discussions

4.1.1 Training Parameters and Time

Table 4.1 describes the training parameters and time per epoch of all the trained neural networks.

Table 4.1: Total Training Parameters for Preexisting and Proposed Models.

CNN Architecture	Number of Parameters
DenseNet 201	18,098,691
GoogLeNet	5,602,979
ResNet 50	23,514,179
ResNet 101	42,504,307
ResNet 152	58,149,955
ResNeXt101	86,748,483
Proposed Model	7,724,523

According to Table 4.1, ResNeXt 101 has the highest number of 86,748,483. The number of parameters indicated that ResNeXt 101 is the most complex model among the CNN architectures, and it has the highest capacity to learn intricate patterns and relationships. However, if the number of data points provided to the model is small, a model with more parameters can also be more prone to overfitting. Many parameters also indicated that ResNeXt 101 could represent complex patterns in the data, allowing the model to capture fine-grained details and nuances of the CT images. However, ResNeXt 101 requires the most extensive memory and computational requirements during training and inference. It requires more memory to store and update the information during training. They also require more computational resources to perform forward and backward passes, which requires more training time to process. The number of parameters dictates the ability to generalise unseen data. A model with few parameters may not have enough capacity to capture complex relationships and cause underfitting. On the other hand, a model with too many parameters may overfit the training data by memorizing specific examples or noise in the data, leading to poor generalisation.

4.1.2 Binary Classification

Various pre-trained architectures were applied to train CT images, including ResNets, GoogLeNet, ResNeXt, and DenseNet. Three distinct depth layers of ResNet, including ResNet 50, ResNet 101, and ResNet 152, were used. ResNeXt 101, DenseNet, and the remaining GoogLeNet were processed. All of the results from the above-mentioned models were documented in this section. There are 3,588 images in each epoch, with a batch size of 64. The training and validation loop was evaluated. The validation loop were reviewed following each training session, and the cycle was then repeated until it reached the maximum epoch. Once the training and validation were completed, the models were evaluated for sensitivity, specificity, precision (PPV), negative predictive value (NPV), accuracy, precision, and F1-score.

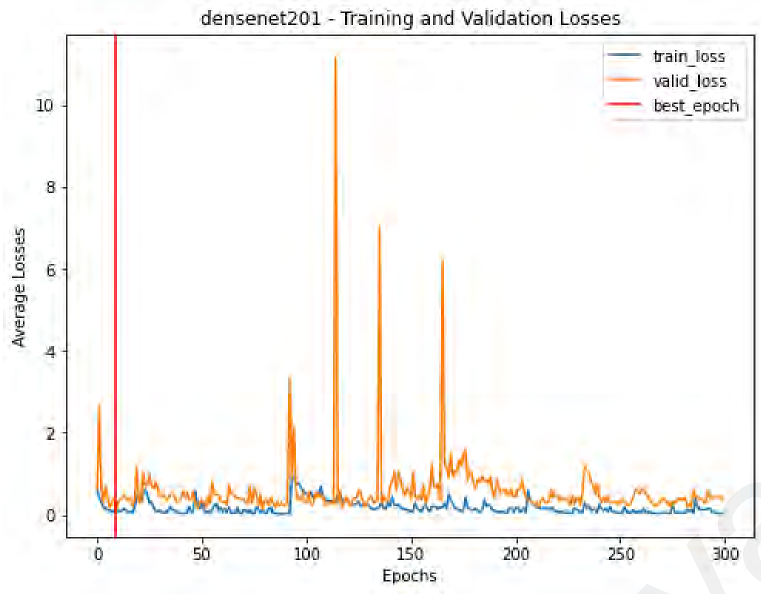


Figure 4.1: DenseNet 201 Training and Validation Loss Binary Classification

DenseNet 201 training loss values vary from 0 to 2, as shown in Figure 4.1. The range of validation loss values is 0 to 10. The best training and validation loss values were found by epoch 9.

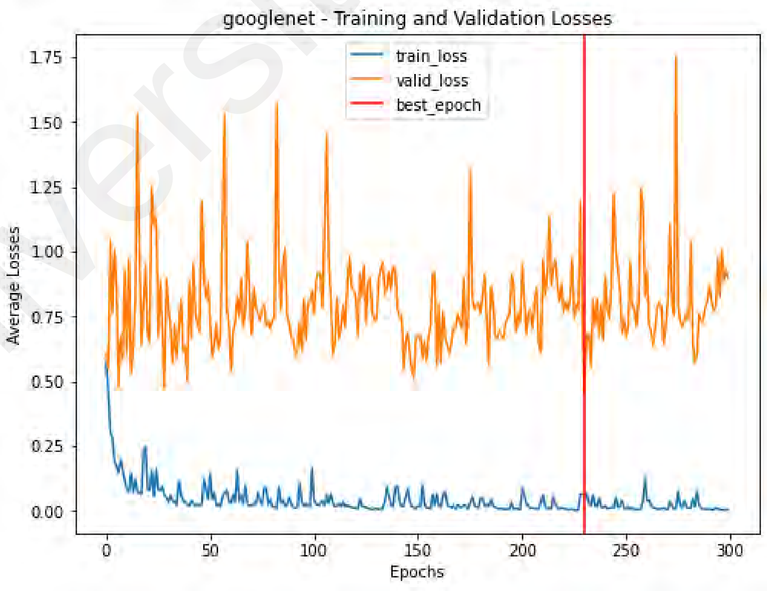


Figure 4.2: GoogLeNet Training and Validation Loss Binary Classification

As shown in Figure 4.2, the training loss values ranged from 0 to 0.5. The range of validation loss values is 0.75 to 1.75. Epoch 230 yielded the optimum settings for training and validation loss.

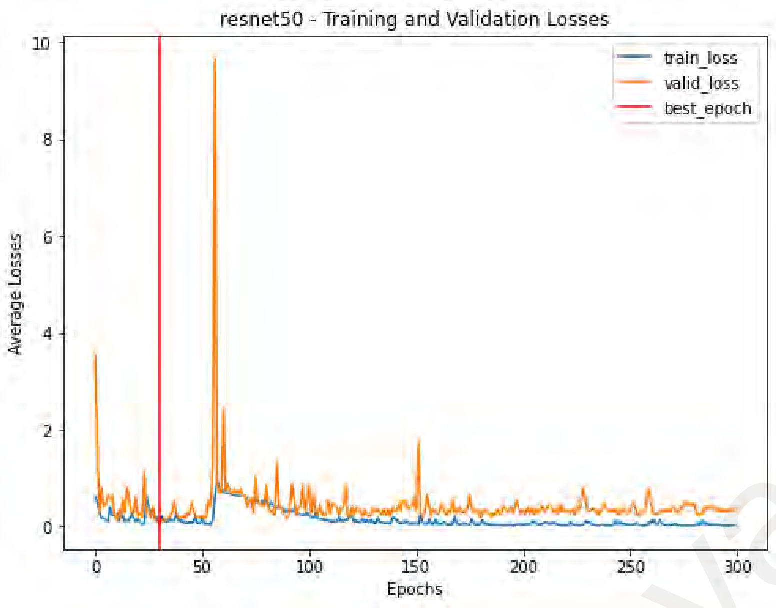


Figure 4.3: ResNet 50 Training and Validation Loss Binary Classification

Variations in training loss levels between 0 and 2 are shown in Figure 4.3. The range of validation loss values is 0 to 10. The best training and validation loss levels were attained by epoch 30.

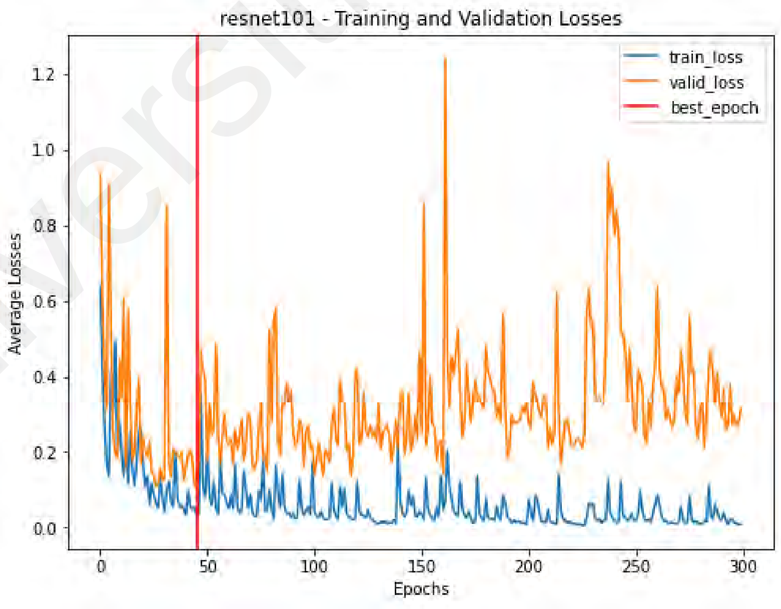


Figure 4.4: ResNet 101 Training and Validation Loss Binary Classification

A training loss value ranging between 0 and 0.7 is shown in Figure 4.4. The values of validation loss vary between 0 and 1.2. Epoch 45 produced the optimal loss levels for both training and validation.

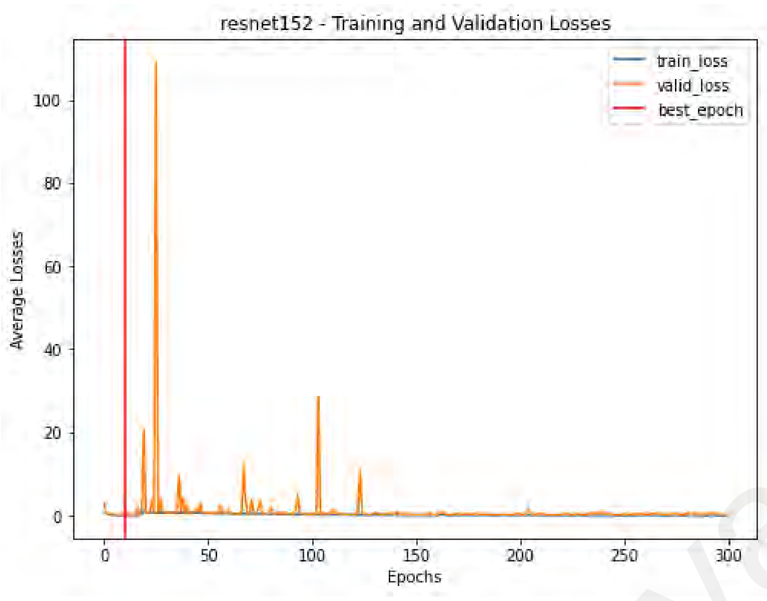


Figure 4.5: ResNet 152 Training and Validation Loss Binary Classification

The training loss value varying between 0 and 10 is shown in Figure 4.5. Between 0 and 110 is the range of validation loss values. The optimal parameters for training and validation loss were obtained using Epoch 10.

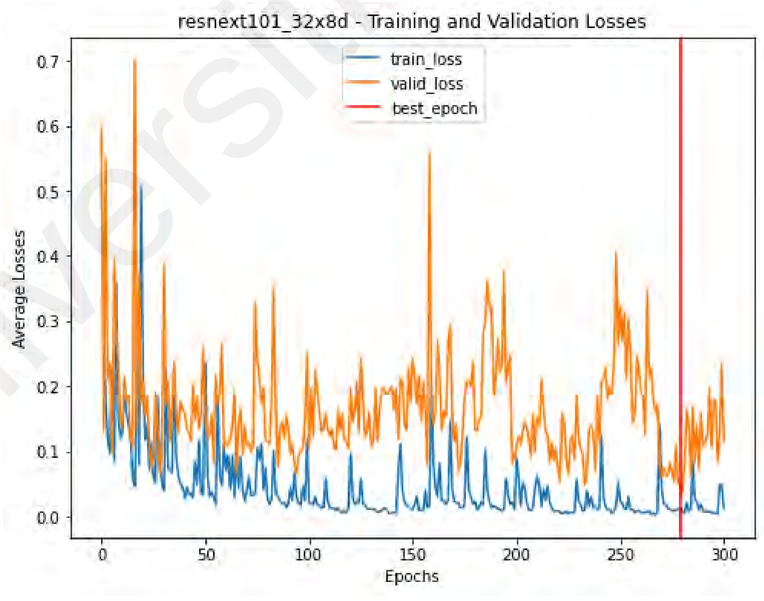


Figure 4.6: ResNeXt 101 Training and Validation Loss Binary Classification

The oscillating training loss value between 0 and 0.5 is shown in Figure 4.6. From 0 to 0.7, the validation loss values vary. Training and validation loss levels were optimal for Epoch 221.

Figures 4.1, 4.2, 4.3, 4.4, 4.5, and 4.6 describe the training and validation losses during transfer learning for binary classification. The blue line in the graph represents the training loss value, and the orange line represents the validation loss value. The red line indicates the best epoch value. The x-axis represents the epochs that were set to 300. The y-axis represents the loss values. The scale of the y-axis varies based on the fluctuations' occurrence. The y-axis for the DenseNet and ResNet 50 scales is from 0 to10 because the highest fluctuation occurred within and slightly above 10. ResNet 152's y-axis scale is from 0 to 100 because the fluctuations happened within, and the peak point happened slightly beyond 100. GoogleNet's y-axis scale is from 0 to 1.75 because the fluctuations happened within and above 1.75. ResNet101's y-axis is from 0 to 1.2 as the fluctuation happened within and slightly beyond 1.2; ResNeXt101's y-axis is from 0 to 0.7 because the fluctuations occurred within the value 0.7.

The graph fluctuation shown in Figures 4.1, 4.2, 4.3, 4.4, 4.5, and 4.6 illustrates how well the model captured the underlying pattern in the data. ResNeXt 101 was seen with the most negligible fluctuation. As the number of parameters increases, the interpretability of the model tends to decrease. ResNeXt 101 displayed the best results compared to the other CNN models. It has the highest number of training parameters and the best performance metrics in all categories. However, deep learning models with many parameters can be prone to overfitting, especially with limited data available for the model to learn. Figure 4.6 indicates ResNeXt 101 achieved the lowest fluctuations magnitude. Throughout the 300 epochs, all the fluctuation peaks occurred within 0.0 to 0.8 average loss values. It depicted the least significant fluctuations, which suggested the model may effectively capture the underlying patterns in the data. There were no signs of overfitting from the graphs that described the models as generalising well to unseen data and memorising the training examples. Hence, it is also the most stable CNN model compared to the rest, as the graph converges towards the end. ResNet 101 and GoogLeNet had insignificant fluctuations between 0 to 1.2 and 0 to 1.8 average loss values, respectively. It suggested that the models learned the underlying patterns in the training data, that their predictions became consistent, and that they converged to a steady state.

ResNet 152 achieved the highest magnitude of fluctuations. It was significant because it exhibited large and abrupt changes in the performance metrics values. The abrupt fluctuations persisted over multiple epochs from 0 to 150, a substantial portion of the training process. DenseNet 201 and ResNet 50 also showed significant fluctuations, but not as abruptly as ResNet 152. DenseNet 201 has a few peaks of fluctuation that are considered sporadic or random, which are less concerning. ResNet 50 also had a few peaks shown within the graph but was inconsistent.

The accuracy of the training and validation of the transfer learning models are depicted below:

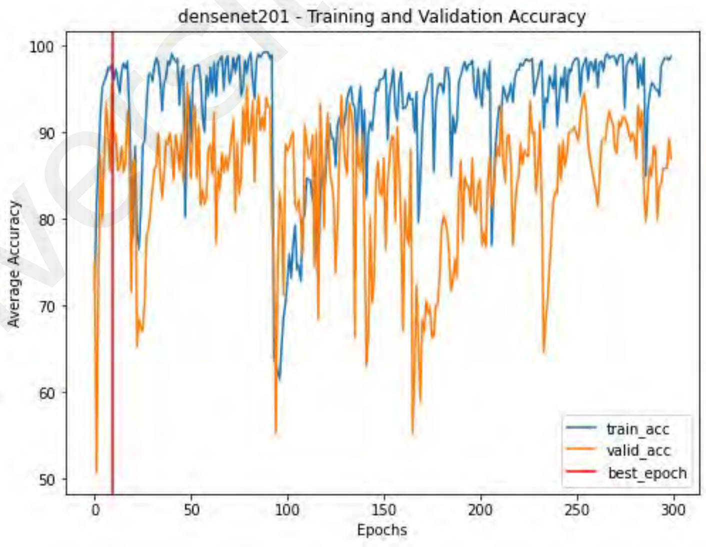


Figure 4.7: DenseNet 201 Training and Validation Accuracy Binary Classification

Figure 4.7 describes the training accuracy values fluctuating from 50 to 100. The validation accuracy fluctuates from 50 to 100. Epoch 9 achieved the best training and validation accuracy values.

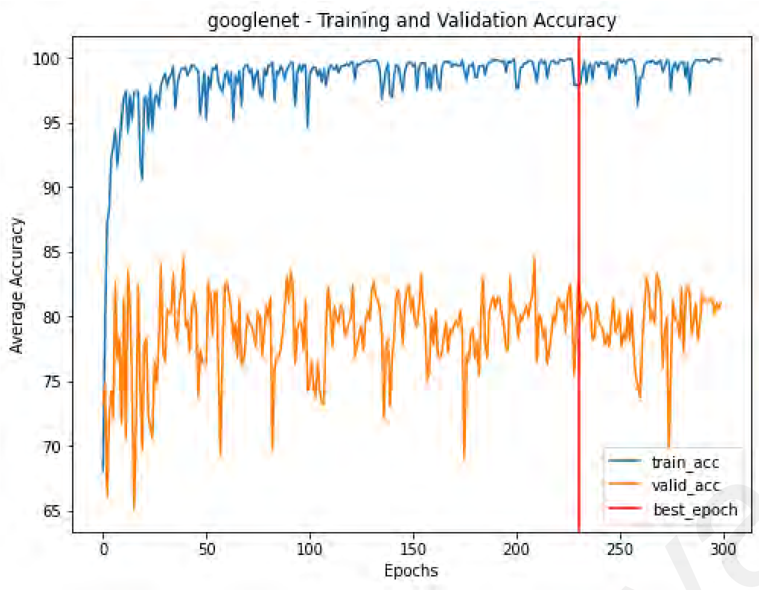


Figure 4.8: GoogLeNet Training and Validation Accuracy Binary Classification

Figure 4.8 describes the training accuracy values fluctuating from 90 – 100. The validation accuracy values fluctuated from 65 to 85. Epoch 230 achieved the best training and validation accuracy values.

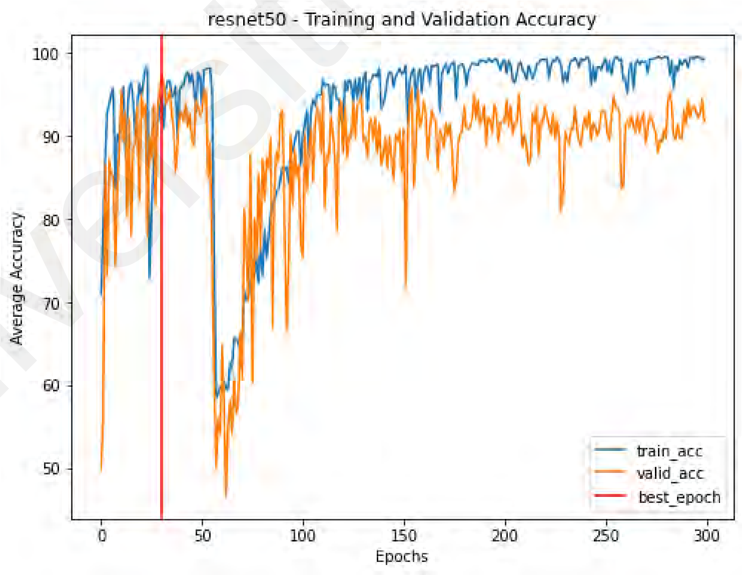


Figure 4.9: ResNet 50 Training and Validation Accuracy Binary Classification

Figure 4.9 shows the training accuracy values fluctuating from 55 to 100. The validation accuracy values fluctuated from 45 to 95. Epoch 30 achieved the best training and validation accuracy values.

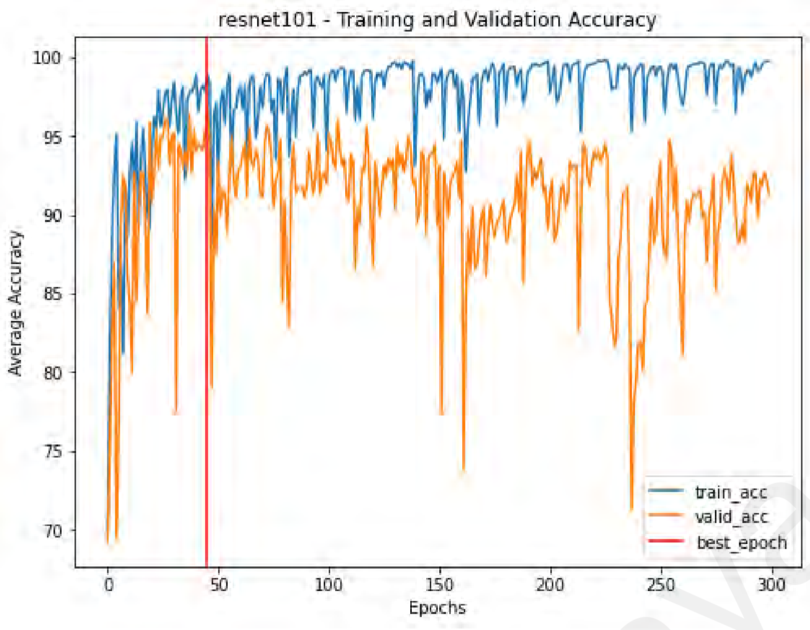


Figure 4.10: ResNet 101 Training and Validation Accuracy Binary Classification

Figure 4.10 indicates the training accuracy values fluctuating from 80 to 100. The validation accuracy values fluctuated from 70 to 98. Epoch 45 achieved the best training and validation accuracy values.

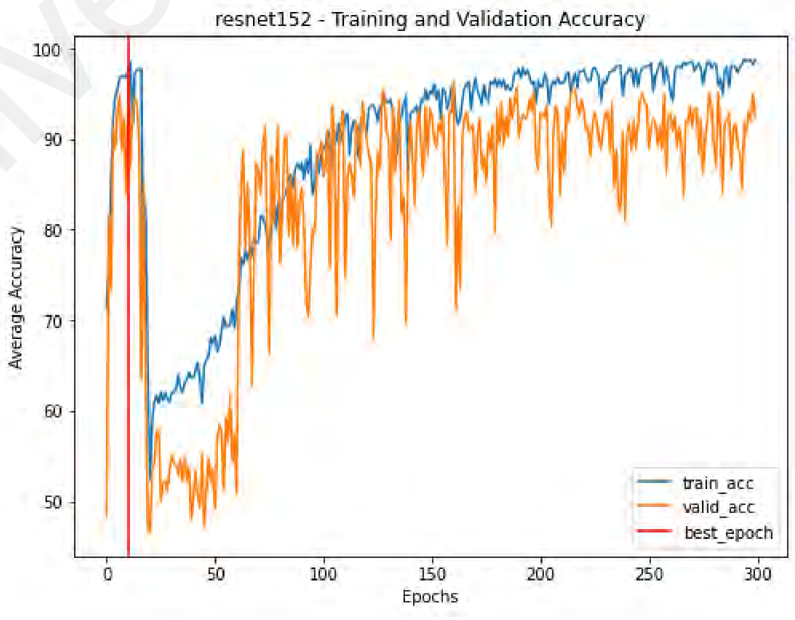


Figure 4.11: ResNet 152 Training and Validation Accuracy Binary Classification

Figure 4.11 indicates the training accuracy fluctuating from 50 to 100. The validation accuracy fluctuated from 45 to 98. Epoch 10 achieved the best training and validation accuracy values.

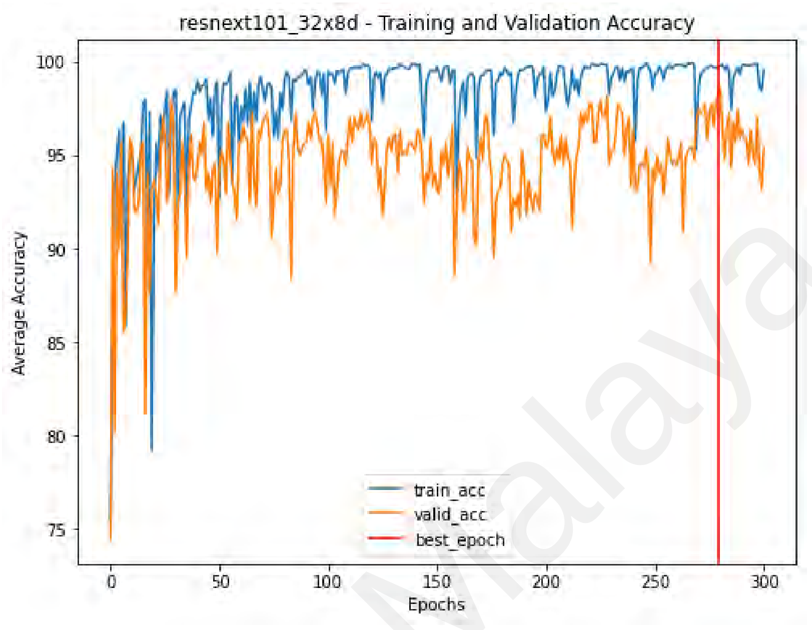


Figure 4.12: ResNeXt 101 Training and Validation Accuracy Binary Classification

Figure 4.12 indicates the training accuracy fluctuating from 80 to 100. The validation accuracy fluctuated from 75 to 98. Epoch 221 achieved the best training and validation accuracy values.

Figures 4.7, 4.8, 4.9, 4.10, 4.11, and 4.12 describe the training and validation accuracy during transfer learning for binary classification for DenseNet, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, and ResNeXt 101. The blue line in the graph represents the training accuracy value, and the orange line represents the validation accuracy value. The red line indicates the best epoch value of the graph. The x-axis represents the epochs that were set to 300. The y-axis represents the accuracy values.

The accuracy graphs were used to evaluate the trend of validation accuracy over epochs. As the model learns, the graph should gradually increase. ResNeXt101 showed that there were low-accuracy discrepancies. The graph's training and validation accuracy are similar in pattern, and the gap is closer, which means the model performed well on the training and generalised unseen data. It described a rising accuracy, indicating that ResNeXt 101 is more proficient at correctly predicting the target classes. In Figure 4.12, ResNeXt 101 validation loss significantly decreased, and the validation accuracy increased. This indicated that the model continued to improve and eventually stabilise.

Figure 4.8 also shows that GoogLeNet's training accuracy is significantly higher than the validation accuracy. It indicated overfitting so that the model may perform well on the training data, but it struggled to generalise unseen data. The graph also showed erratic behaviour. It suggested that the model may have reached its performance limit or encountered difficulties capturing specific patterns.

DenseNet 201, ResNet 50, and ResNet 152 showed that training and validation gradually increased, indicating that the models initially captured the patterns and improved their prediction. However, during training and validation, the models started to overfit and became too specialised in the training examples. It memorised specific details and noises present in the training set. As a result, the models' performances on the validation set started to decline. The sudden drop present in the training and validation accuracies indicated that the models were memorising the training data too closely, leading to poor generalisation. During that period, the models could not effectively capture the underlying patterns and relationships in the data. After the dip, the models continued to improve, and the accuracy gradually increased. The phenomenon suggested that the model started to generalise better and learn more meaningful features from the data. It adjusted the effect to reduce the memorisation effect and focus on capturing more relevant information.

ResNet 101 training and validation accuracy depicted sharp and erratic fluctuations that suggested the model's instability. The fluctuations indicated that the model struggled to capture the data. The graph showed that the training line slowly reduced its fluctuations and increased as it approached the end of the epoch. However, the validation line continued with sharp, erratic behaviour and gradually decreased. The phenomenon suggested that the model struggled to converge to an optimal solution. Hence, it became too special in the training data and failed to generalise unseen data.

4.1.2.1 Confusion Matrix

After completing the training and validation set, the confusion matrix was applied to each trained neural network, plotting Class 0 as 'normal' with 40 images and Class 1 as 'COVID-19 pneumonia' with 35 images. The confusion matrix is a 2 x 2 table that summarizes the performance of a binary classification model on a test dataset. The test images were divided into the following prediction results: True Positives (TP) are the number of incidents that predict Class 1 correctly; True Negatives (TN) are the number of occurrences that correctly predict Class 0; False Positives (FP) are the number of instances that incorrectly predict Class 1 when the model should predict it as Class 0; and False Negatives (FN) are the number of instances that incorrectly categorise Class 0 when it actually should be categorised under Class 1. The values from the confusion matrix were calculated to perform performance metrics.

Figure 4.13 describes the confusion matrix for DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, and ResNeXt 101.

DenseNet201	0	1
0	33	4
1	2	36

GoogLeNet	0	1
0	28	11
1	7	29

ResNet 50	0	1
0	30	2
1	5	38

ResNet 101	0	1
0	31	0
1	4	40

ResNet 152	0	1
0	33	1
1	2	39

ResNeXt101	0	1
0	33	0
1	2	40

Figure 4.13: Binary Classification Confusion Matrix

Figure 4.13 shows that DenseNet 201 had two false positives indicating that the model incorrectly predicted two Class 1s, which should belong to Class 0s, and four false negatives indicating the CNN model mistakenly predicted four Class 0s, which should be Class 1s.

GoogLeNet achieves eleven false positives indicating that the model incorrectly predicted eleven Class 1s, which should be under Class 0, and seven false negatives indicating that the model mistakenly predicted seven Class 0s, which should be categorised under Class 1.

ResNet 50 obtains two false positives, indicating that the model identified two images under Class 1, which should be under Class 0. It also obtained five false negatives, indicating that the model identified five images under Class 0, which should be under Class 1.

ResNet 101 obtains zero false positives, indicating that the model correctly identified all Class 0 images. It also obtained four false negatives, indicating that the model identified four images under Class 0, which should be under Class 1.

ResNet 152 obtains one false positive, indicating that the model categorises one image under Class 1, which should be under Class 0. It obtains two false negatives, indicating the model identified two images under Class 0, which should be under Class 1.

ResNeXt 101 obtains zero false positives, indicating that the model correctly identified all Class 0 images. It also obtained two false negatives, indicating the model incorrectly identified two images under Class 0, which should be under Class 1.

In short, ResNeXt 101 has two wrong classifications, ResNet 152 has three incorrect types, and ResNet 101 has four misclassifications. GoogLeNet had the most, with eleven misclassifications.

4.1.2.2 Performance Metrics

After training and validation, the models were tested using an unseen dataset. The accuracy (ACC), sensitivity (TPR), specificity (TNR), PPV, NPV, and F1 scores were recorded. Table 4.2 shows the binary classification performance metrics for DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, and ResNeXt 101. Figure 4.14 is the graph derived from Table 4.2.

Table 4.2: Test Set Performance Metrics.

CNN Models	TPR (%)	TNR (%)	PPV (%)	NPV (%)	ACC (%)	F1 Score (%)
DenseNet201	93.81	90.00	89.14	94.32	91.78	91.42
GoogLeNet	80.95	72.50	72.03	81.31	76.44	76.23
ResNet 50	86.19	95.83	94.76	88.80	91.33	90.27
ResNet 101	88.57	99.58	99.47	90.88	94.44	93.70
ResNet 152	90.00	98.33	97.93	91.83	94.44	93.80
ResNeXt101	95.71	100.00	100.00	96.39	98.00	97.81

Table 4.2, ResNeXt 101, showed a TPR of 95.71% with two misclassifications of 'COVID-19 pneumonia' images. Whereas DenseNet 201 had a TPR of 93.81% and ResNet152 attained a TPR of 90%, both with two wrong COVID-19 images classified. ResNet 101 attained a TPR of 88.57% with four false classifications, and ResNet 50 gained 86.19% with five incorrect categories of COVID-19 images. Lastly, GoogLeNet had a TPR of 80.95% with seven wrong COVID-19 image classifications.

PPV, also known as precision, indicated how often the model predicted 'COVID-19 pneumonia' was correct. PPV considered the models to be performing well. Suppose the precision value was not good despite the accuracy of the models. The models would not be considered reasonable. ResNeXt 101 achieved 100% PPV, ResNet 101 achieved 99.47%, followed by ResNet152, which achieved 97.93%. GoogLeNet achieved the lowest precision of 72.03%, with seven wrong classifications of 'normal'.

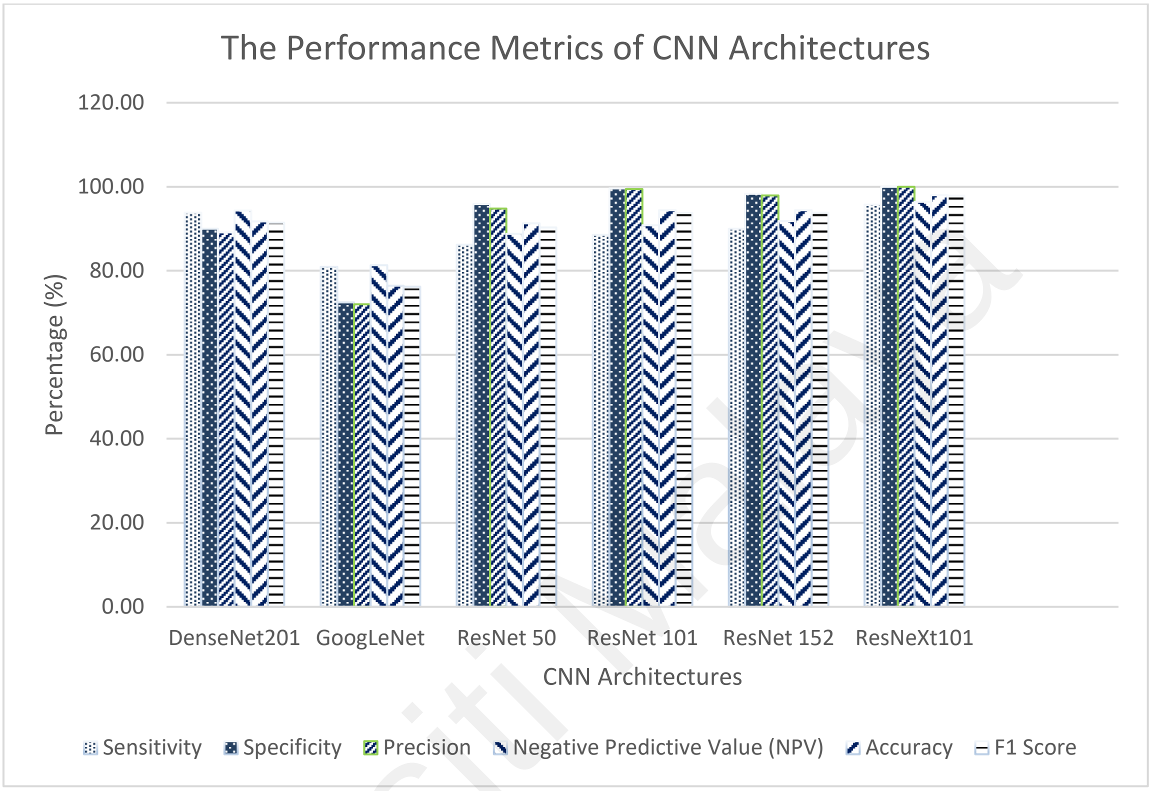


Figure 4.14: Binary Classification Performance Metrics Graph

Figure 4.14 depicts ResNeXt 101 as having the highest performance metrics values, followed by ResNet 152 and ResNet 101. GoogLeNet achieved the lowest performance metrics values.

Figure 4.14 describes the binary classification performance metrics graph, showing ResNeXt 101, ResNet 101, and ResNet 152 performed better in classifying non-COVID-19 images. ResNeXt 101 had the highest accuracy and F1 score of 98% and 97.81%, followed by ResNet 152 with an accuracy of 94.44% and an F1 score of 93.80%. ResNet 101 achieved an accuracy of 94.44% and an F1 score of 93.70%. DenseNet had 91.78% accuracy and a 91.42% F1 score. ResNet 50's accuracy and F1 Score were 91.33% and

90.27%, and the lowest accuracy and F1 Score was GoogLeNet, with 76.44% and 76.23%, respectively.

4.1.3 Multi-class Classification

The CT images were trained using existing CNN architectures: ResNets, GoogleNet, ResNeXt, and DenseNet. ResNet was performed thrice with different depth layers, such as ResNet 50, ResNet 101, and ResNet 152. The dataset was trained with the proposed model.

Each epoch has 5,846 images with a batch size of 16. The batch size for multi-class classification is smaller than for binary classification. We used Google Colab Pro version 2021 to train, validate, and test for binary classification. Google Colab provided users with access to 10 GB to 25 GB of GPU memory. However, Google Colab has created an additional tier called Google Colab Pro+, which is costly. Hence, we have switched to using Kaggle. Kaggle's Video Random Access Memory (VRAM) is less than the Google Colab Pro 2021 version. Since the batch size determines the number of samples processed in each iteration during training, larger batch sizes require more memory to store the intermediate activations and gradients for backpropagation. With a larger batch size, the training process was more efficient, leading to faster convergence. However, in the study for multi-class classification, we applied a smaller batch size because the computational resources were limited. Once the training and validation were completed, the models were tested using the confusion matrix and performance metrics.

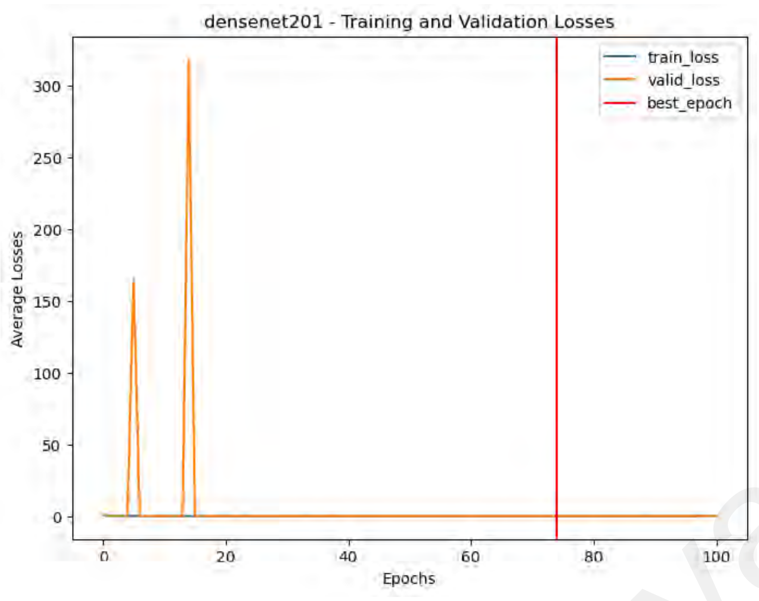


Figure 4.15: DenseNet 201 Training and Validation Loss Multi-class Classification

Figure 4.15 DenseNet201 training loss values maintained constant from 0 to 10. The validation loss values fluctuate from 0 to 300. Epoch 75 obtained the best training and validation loss values.

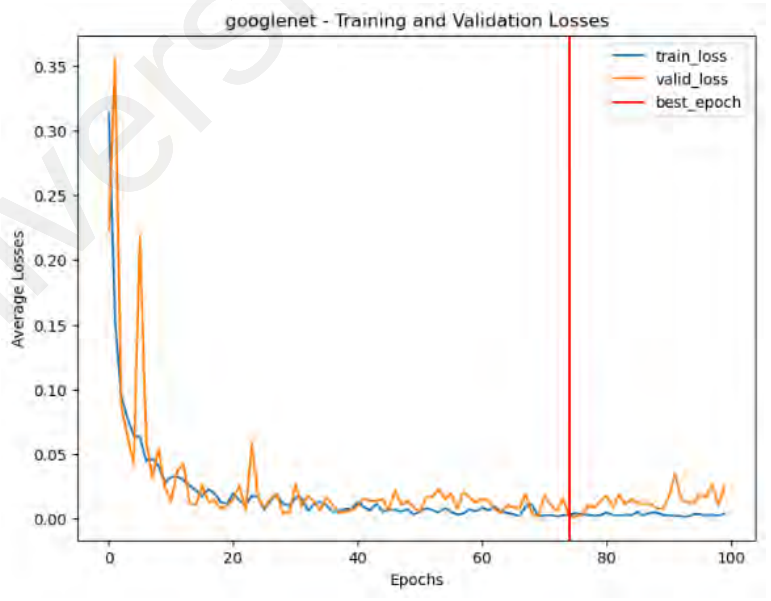


Figure 4.16: GoogLeNet Training and Validation Loss Multi-class Classification

Figure 4.16 GoogLeNet training loss values maintained constant from 0 to 0.10. The validation loss values fluctuate from 0 to 0.35. Epoch 75 obtained the best training and validation loss values.

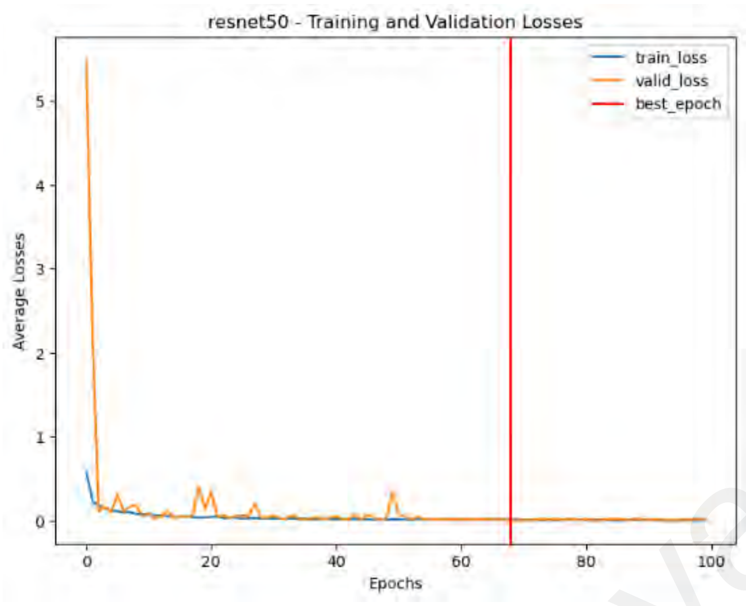


Figure 4.17: ResNet 50 Training and Validation Loss Multi-class Classification

Figure 4.17 indicates the training loss values maintained constant from 0 to 1. The validation loss values fluctuate from 0 to 5. Epoch 69 obtained the best training and validation loss values.

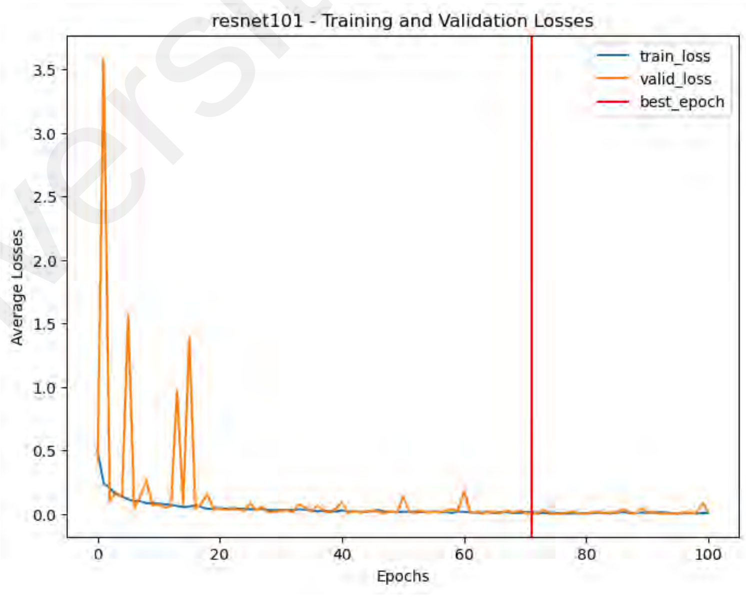


Figure 4.18: ResNet 101 Training and Validation Loss Multi-class Classification

Figure 4.18 shows the training loss values maintained constant from 0 to 0.5. The validation loss values fluctuate from 0 to 3.5. Epoch 72 obtained the best training and validation loss values.

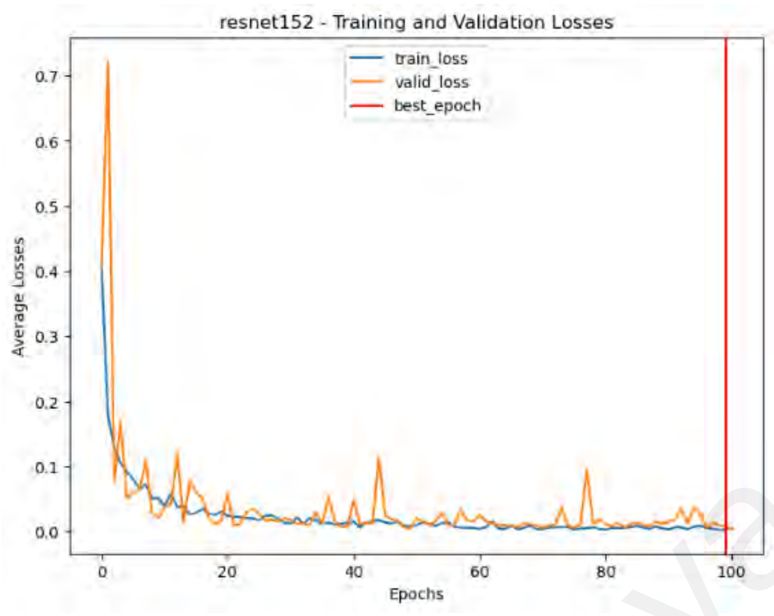


Figure 4.19: ResNet 152 Training and Validation Loss Multi-class Classification

Figure 4.19 depicts the training loss values maintained constant from 0 to 0.1. The validation loss values fluctuate from 0 to 0.8. Epoch 100 obtained the best training and validation loss values.

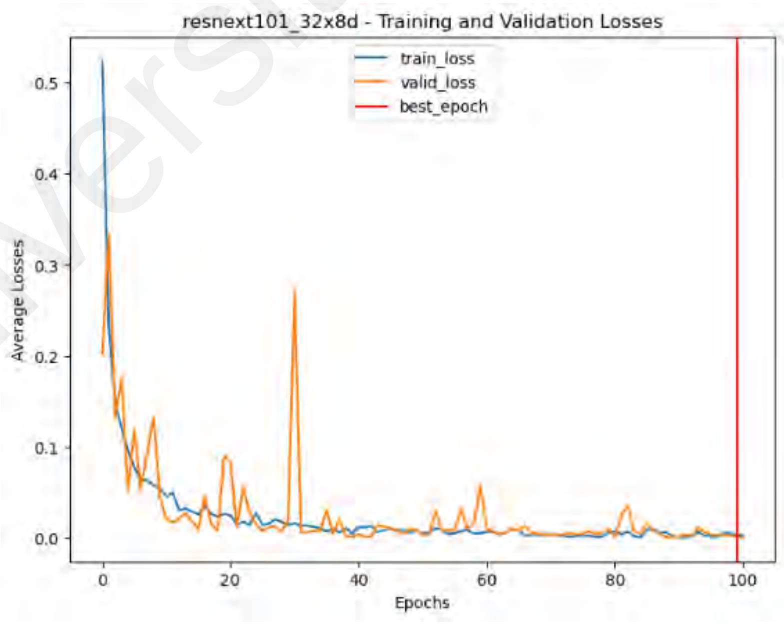


Figure 4.20: ResNeXt 101 Training and Validation Loss Multi-class Classification

Figure 4.20 shows that the training loss values maintained constant from 0 to 0.1. The validation loss values fluctuate from 0 to 0.5. Epoch 100 obtained the best training and validation loss values.

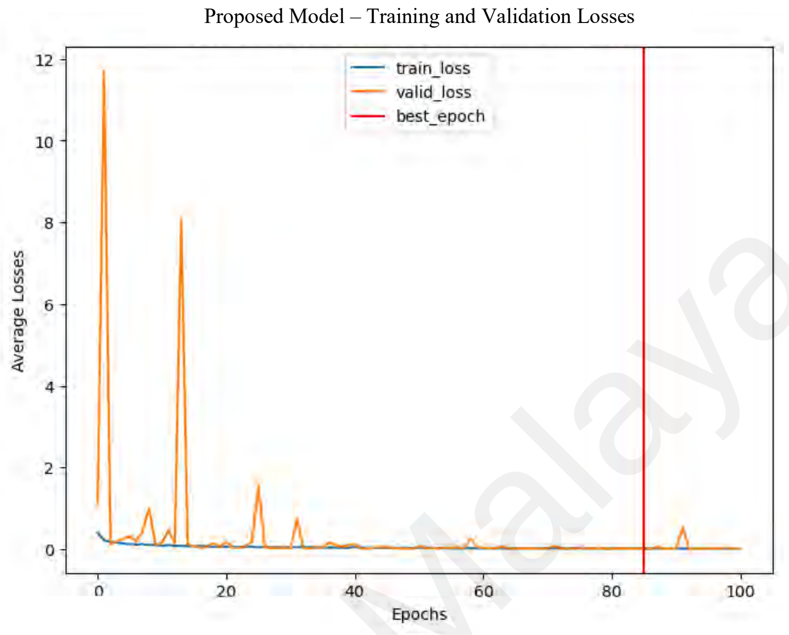


Figure 4.21: Proposed Model Training and Validation Loss Multi-class Classification

Figure 4.21 describes the training loss values maintained constant from 0 to 2. The validation loss values fluctuate from 0 to 12. Epoch 86 obtained the best training and validation loss values.

Figures 4.15, 4.16, 4.17, 4.18, 4.19, 4.20, and 4.21 describe the training and validation losses during transfer learning for multi-class classification for DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, ResNeXt 101, and the proposed model. The blue line in the graph represents the training loss value, and the orange line represents the validation loss value. The red line indicates the best epoch value. The x-axis represents the epochs that were set to 100. The y-axis represents the loss values. The scale of the y-axis varies based on the fluctuations' occurrence. The graphs provided insights into the performance of the multi-class classifications. The loss values represent the dissimilarities between the predicted class probabilities and the true class labels throughout the training epochs. The x-axis represents the epoch number, while the y-axis represents the loss values. ResNeXt 101, GoogLeNet, and ResNet 152 showed similar validation loss patterns.

Initially, the graphs exhibited fluctuations as the models started to learn. The fluctuations were insignificant, as the validation set may contain inherent noise or variation. As the models progressed, a steady decrease in the validation loss suggested that the models successfully generalised and made accurate predictions on unseen data. The steady decrease suggested that the models captured relevant features and patterns that were not specific to the training data. The models did not show signs of overfitting or underfitting. ResNet 50 showed a steady decrease in training and validation loss, which suggested that the model successfully generalised and made accurate predictions of unseen data. ResNet 101 and the proposed model initially exhibit fluctuations as the models were randomly initialised, and their predictions were inaccurate. As the training and validation progressed, a general decreasing trend in the validation loss occurred. All three models did not show any signs of overfitting. During the training process, DenseNet 201 displayed significant fluctuations, with higher values in the first 20 epochs. The phenomenon indicated that the model's parameters were randomly initialised and its predictions were inaccurate. As the training progressed, the model learned to capture the underlying patterns in the data, and a decreasing trend in validation loss occurred.

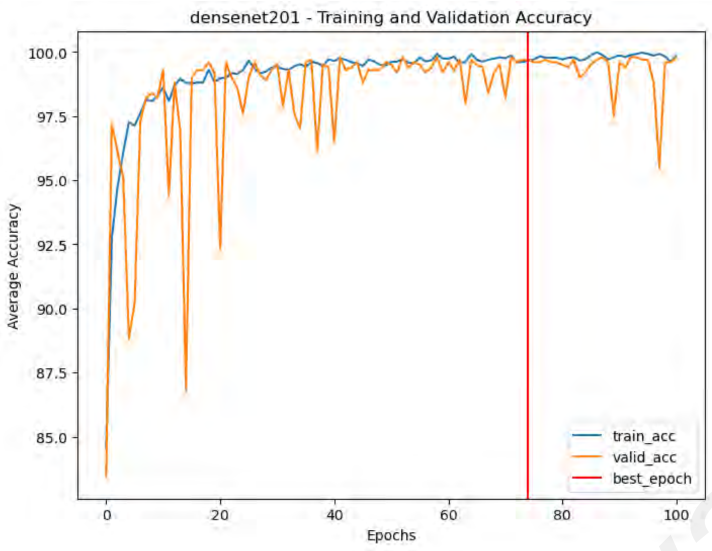


Figure 4.22: DenseNet201 Training and Validation Accuracy Multi-class Classification

Figure 4.22 describes the training accuracy values maintained constant from 95 to 100. The validation accuracy values fluctuate from 85 to 100. Epoch 75 obtained the best training and validation accuracy values.

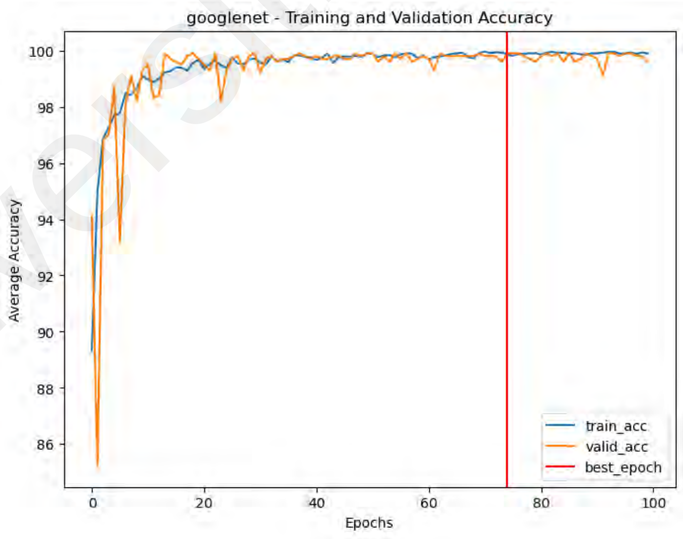


Figure 4.23: GoogLeNet Training and Validation Accuracy Multi-class Classification

Figure 4.23 describes the training accuracy values maintained constant from 96 to 100. The validation accuracy values fluctuate from 86 to 100. Epoch 75 obtained the best training and validation accuracy values.

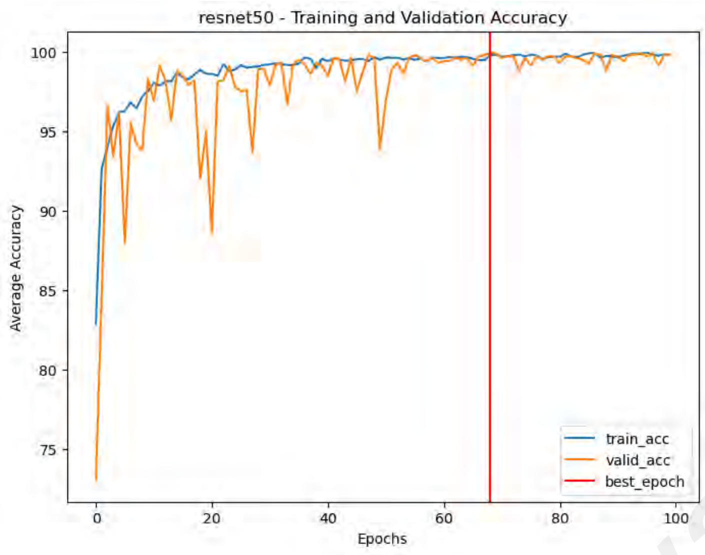


Figure 4.24: ResNet 50 Training and Validation Accuracy Multi-class Classification

Figure 4.24 describes the training accuracy values maintained constant from 95 to 100. The validation accuracy values fluctuate from 85 to 100. Epoch 69 obtained the best training and validation accuracy values.

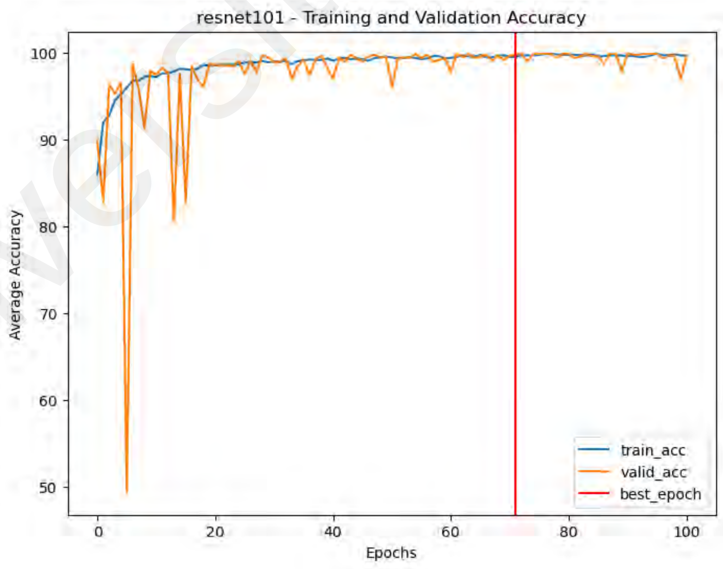


Figure 4.25: ResNet 101 Training and Validation Accuracy Multi-class Classification

Figure 4.25 describes the training accuracy values maintained constant from 85 to 100. The validation accuracy values fluctuate from 50 to 100. Epoch 72 obtained the best training and validation accuracy values.

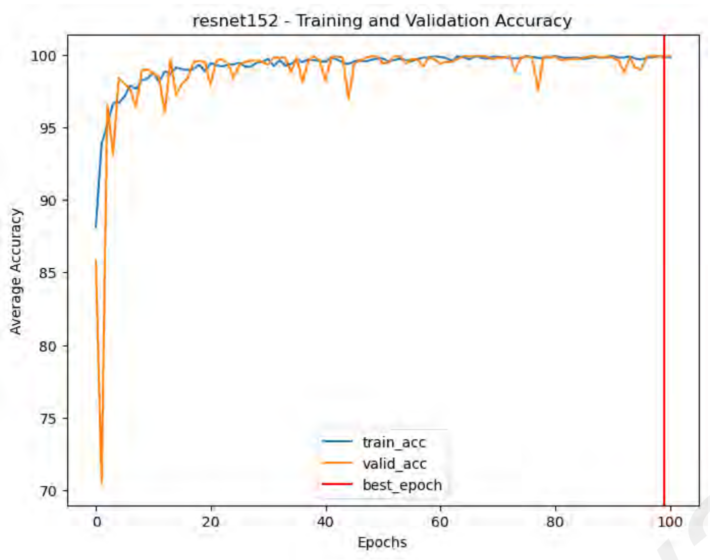


Figure 4.26: ResNet 152 Training and Validation Accuracy Multi-class Classification

Figure 4.26 describes the training accuracy values maintained constant from 90 to 100. The validation accuracy values fluctuate from 70 to 100. Epoch 100 obtained the best training and validation accuracy values.

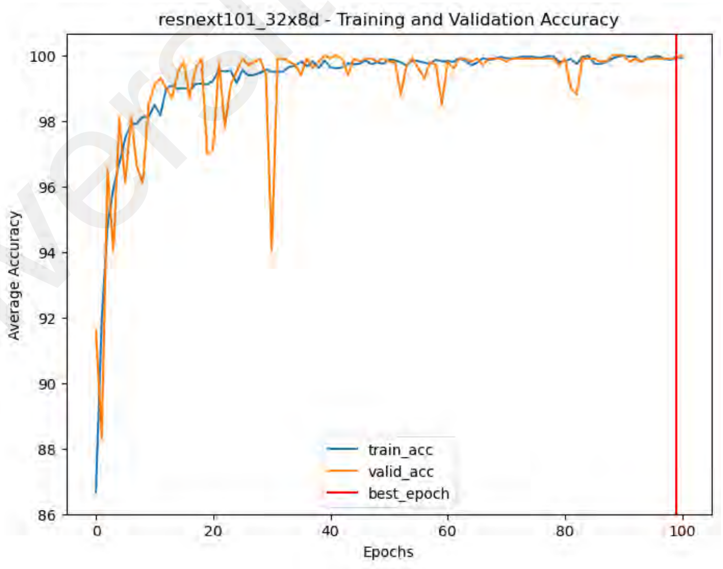


Figure 4.27: ResNeXt Training and Validation Accuracy Multi-class Classification

Figure 4.27 describes the training accuracy values maintained constant from 96 to 100. The validation accuracy values fluctuate from 86 to 100. Epoch 100 obtained the best training and validation accuracy values.

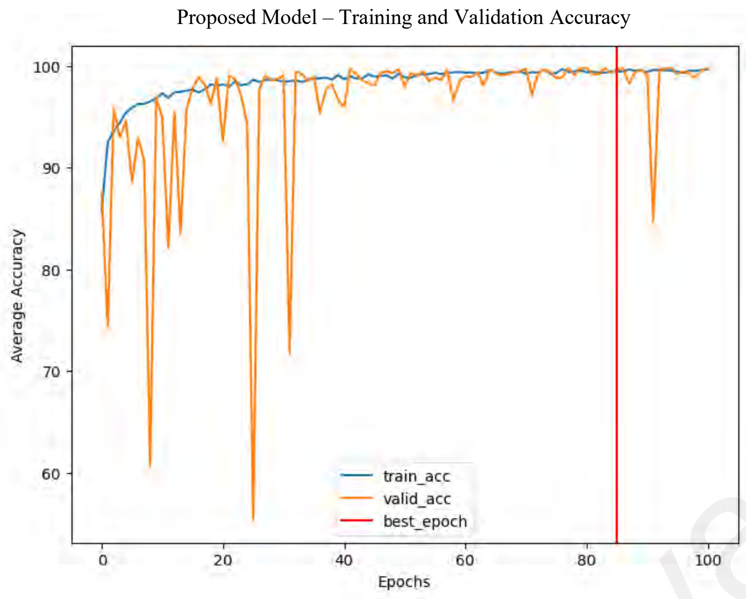


Figure 4.28: Proposed Model Training and Validation Accuracy Multi-class Classification

Figure 4.28 describes the training accuracy values maintained constant from 80 to 100. The validation accuracy values fluctuate from 60 to 100. Epoch 86 obtained the best training and validation accuracy values.

Figures 4.22, 4.23, 4.24, 4.25, 4.26, 4.27, and 4.28 indicate the training and validation accuracies during transfer learning for multi-class classification for DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, ResNeXt 101, and the proposed model. The blue line in the graph represents the training accuracy value, and the orange line represents the validation accuracy value. The red line indicates the best epoch value. The x-axis represents the epochs that were set to 100. The y-axis represents the accuracy values. The scale of the y-axis varies based on the fluctuations' occurrence. The training accuracy graph represented the proportion of correctly classified data samples from the training set. The x-axis represents the epoch number, while the y-axis represents the accuracy values. Initially, the training accuracy started with relatively low values as the parameters began to be initialised. The graph shows a gradual increase in training accuracy as the training progresses. The phenomenon indicated that the models learned and captured underlying patterns in the training data.

Figure 4.27 and Figure 4.23 show that ResNeXt 101 and GoogLeNet performed the best in training and validation accuracy. Both models also described similar validation accuracy patterns. ResNeXt 101 and GoogLeNet's validation accuracy started low with insignificant fluctuations as the models' parameters were randomly initialised. The validation graph progressed as their accuracies continued to increase, which means the models managed to generalise well and make accurate predictions. It also suggested that relevant features and patterns not specific to the training set were accurately recognised. The fluctuations in the validation graphs were minimal, and the graph did not stagnate or decrease, while the training accuracy increase indicated that the models successfully generalised new and unseen data.

Figures 4.22, 4.24, 4.25, 4.26 and 4.28 show that DenseNet 201, ResNet 50, ResNet 101, ResNet 152, and the proposed model were observed with more prominent fluctuations throughout the validation accuracy. At the initial stage of the validation accuracy graph, DenseNet 201 showed that the fluctuations were erratic, but the accuracy values did not show sharp drops. However, the fluctuations often occurred throughout the epoch and intensified towards the end of the graph. The phenomenon indicated that the model lacks stability, and the model's performance varies significantly across different evaluation samples or even between consecutive epochs.

ResNet 50, ResNet 101, and ResNet 152 showed that although there were sudden fluctuations initially, the validation accuracy increased and stabilised throughout the epoch. Small fluctuations in the validation accuracy are expected, as the validation set may contain noise or variation. Therefore, the fluctuations overall were insignificant to consider as signs of the model overfitting.

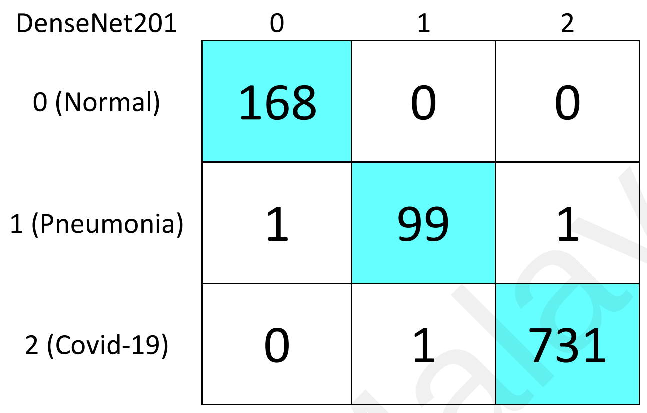
The validation accuracy of the proposed model displayed sharp and erratic fluctuations during the early stages of the graph. The sharp drops and erratic patterns happened during the first 30 epochs of the graph. The phenomenon is considered normal as the dataset may contain inherent noise or variations. As the validation accuracy progressed, it gradually increased and stabilised with very few fluctuations, which showed that it could generalise underlying patterns and unseen data.

4.1.3.1 Confusion Matrix

Once the training and validation were completed, the study tested the images with the existing and proposed models. For multi-class classification, confusion matrices were used for validation and testing. The elaboration of the confusion matrix for multi-class classification requires a comprehensive explanation of different components and metrics obtained from the confusion matrix. The classes were categorised into '0' for normal lung conditions' CT images, '1' for pneumonia patients' CT images, and '2' for COVID-19 patients' CT images.

Figures 4.29, 4.30, 4.31, 4.32, 4.33, 4.34, and 4.35 are the validation set confusion matrix results. The figures indicate that the diagonal elements of the confusion matrix represent the instances that the model correctly predicted for each class. For Class 0, the value in the (0, 0) position represents the number of instances that belong to Class 0 and are correctly identified as Class 0. For Class 1, the value in the (1, 1) position represents the number of times the model predicted the images belong to Class 1 and are correctly predicted. For Class 2, the value in the (2, 2) position represents the number of times the model predicted the images belong to Class 2 and are correctly predicted.

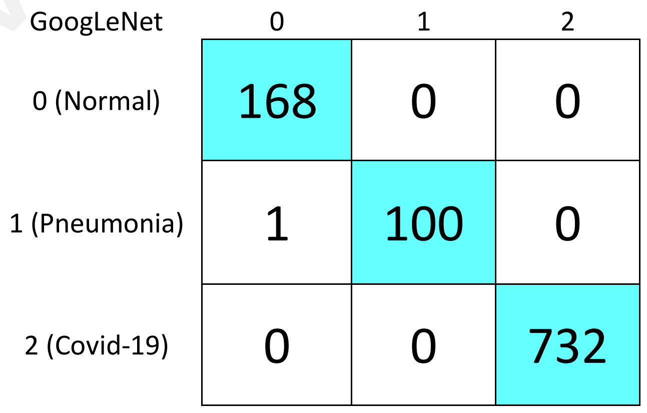
The off-diagonal entries are the false positives and false negatives; they represent the number of instances in the model of misclassified images. For Class 0, the values in the (0, 1) and (0, 2) positions represent the number of instances that belong to Class 0 but were predicted as Class 1 and Class 2, respectively. For Class 1, the values in the (1, 0) and (1, 2) positions represent the number of times that belong to Class 1 but were categorised as Class 0 and Class 2, respectively. For Class 2, the values in the (2, 0) and (2, 1) positions represent the number of times that belong to Class 2 but were identified as Class 0 and Class 1 instead.



DenseNet201	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	1	99	1
2 (Covid-19)	0	1	731

Figure 4.29: DenseNet 201 Validation Set Confusion Matrix

Figure 4.29 indicates that the model identified Class 0 perfectly without misclassification. It misclassified one image of Class 1 as Class 0 and another as Class 1 as Class 2. It also misclassified one image of Class 2 as Class 1. It made a total of three misclassifications.



GoogLeNet	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	1	100	0
2 (Covid-19)	0	0	732

Figure 4.30: GoogLeNet Validation Set Confusion Matrix

Figure 4.30 indicates that the model identified Class 0 perfectly without misclassification. It misclassified one image of Class 1 as Class 0. It also correctly classified all Class 2 images, making a total of one misclassification.

ResNet 50	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	0	101	0
2 (Covid-19)	0	1	731

Figure 4.31: ResNet 50 Validation Set Confusion Matrix

Figure 4.31 indicates that the model identified Class 0 perfectly without misclassification. It correctly classified all images of Class 1. It also misclassified one image of Class 2 as Class 1. It made a total of one misclassification.

ResNet 101	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	0	100	1
2 (Covid-19)	0	0	732

Figure 4.32: ResNet 101 Validation Set Confusion Matrix

Figure 4.32 indicates that the model identified Class 0 perfectly without misclassification. It misclassified one image of Class 1 as Class 2. It also correctly classified all images in Class 2. It made a total of one misclassification.

ResNet 152	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	0	100	1
2 (Covid-19)	0	0	732

Figure 4.33: ResNet 152 Validation Set Confusion Matrix

Figure 4.33 indicates that the model identified Class 0 perfectly without misclassification. It misclassified one image of Class 1 as Class 2. It also correctly classified all images of Class 2. It made a total of one misclassification.

ResNeXt101	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	0	101	0
2 (Covid-19)	0	0	732

Figure 4.34: ResNeXt 101 Validation Set Confusion Matrix

Figure 4.34 indicates that the model identified Class 0 perfectly without misclassification. It correctly classified all images of Class 1. It also correctly classified all images of Class 2. No misclassifications were made.

Proposed Model	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	0	100	1
2 (Covid-19)	1	0	731

Figure 4.35: Proposed Model's Validation Set Confusion Matrix

Figure 4.35 indicates that the model identified Class 0 perfectly without misclassification. It misclassified one image of Class 1 as Class 2. It also misclassified one image of Class 2 as Class 0. It made a total of two misclassifications.

Figures 4.36, 4.37, 4.38, 4.39, 4.40, 4.41 and 4.42 indicate the test set confusion matrix results.

DenseNet201	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	3	98	0
2 (Covid-19)	0	0	732

Figure 4.36: DenseNet 201 Test Set Confusion Matrix

Figure 4.36 indicates that the model identified Class 0 perfectly without misclassification. It misclassified three images of Class 1 as Class 0. It correctly classified all images of Class 2. It made a total of three misclassifications.

GoogLeNet	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	2	99	0
2 (Covid-19)	0	0	732

Figure 4.37: GoogLeNet Test Set Confusion Matrix

Figure 4.37 indicates that the model identified Class 0 perfectly without misclassification. It misclassified two images of Class 1 as Class 0. It correctly classified all images of Class 2. It made a total of two misclassifications.

ResNet 50	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	0	101	0
2 (Covid-19)	0	0	732

Figure 4.38: ResNet 50 Test Set Confusion Matrix

Figure 4.38 indicates that the model identified Class 0 perfectly without misclassification. It correctly classified all images of Class 1. It also correctly classified all images of Class 2. No misclassification was made.

ResNet 101	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	2	99	0
2 (Covid-19)	0	0	732

Figure 4.39: ResNet 101 Test Set Confusion Matrix

Figure 4.39 indicates that the model identified Class 0 perfectly without misclassification. It misclassified two images of Class 1 as Class 0. It correctly classified all images of Class 2. It made a total of two misclassifications.

ResNet 152	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	1	98	2
2 (Covid-19)	0	0	732

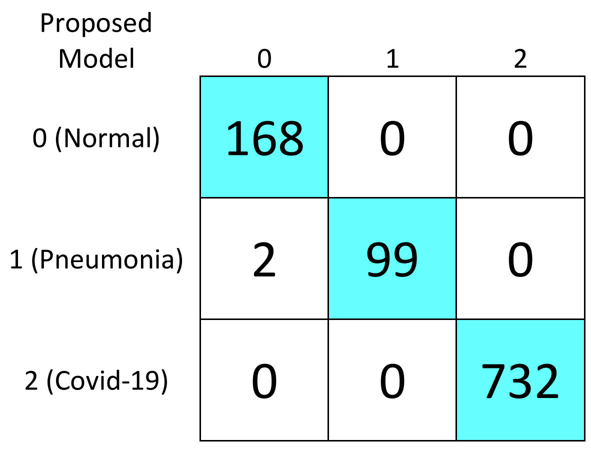
Figure 4.40: ResNet 152 Test Set Confusion Matrix

Figure 4.40 indicates that the model identified Class 0 perfectly without misclassification. It misclassified one image of Class 1 as Class 0 and two of Class 1 as Class 2. It correctly classified all images of Class 2. It made a total of three misclassifications.

ResNeXt101	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	2	99	0
2 (Covid-19)	0	0	732

Figure 4.41: ResNeXt 101 Test Set Confusion Matrix

Figure 4.41 indicates that the model identified Class 0 perfectly without misclassification. It misclassified two images of Class 1 as Class 0. It correctly classified all images of Class 2. It made a total of two misclassifications.



Proposed Model	0	1	2
0 (Normal)	168	0	0
1 (Pneumonia)	2	99	0
2 (Covid-19)	0	0	732

Figure 4.42: Proposed Model's Test Set Confusion Matrix

Figure 4.42 indicates that the model identified Class 0 perfectly without misclassification. It misclassified two images of Class 1 as Class 0. It correctly classified all images in Class 2. It had a total of two misclassifications.

The multi-class confusion matrix does not have positive or negative classes. Therefore, it was confusing to determine the performance metrics compared to binary classification. However, we can calculate and measure the other classes. The approach was to identify the positive and negative classes within the multi-class classification and break them into binary problems using one-to-one or one-to-all. The method used in the study is one-to-all, where Class 0 is considered the positive label and the rest are negative labels. Using the one-to-all strategy lowers the computational costs because only one class was considered positive and the rest were negative labels.

4.1.3.2 Performance Metrics

The multiclass classification performance metrics were harder to achieve. The method used was the one-to-all strategy to convert multiclass problems into a series of binary tasks for each class in the target. Hence, classifying the CT image types can be binarised into 4 tasks: 1) ‘normal’ CT images versus ‘pneumonia’ and COVID-19,; 2) ‘pneumonia’ CT images versus ‘normal’ and ‘COVID-19 pneumonia’,; and 3) ‘COVID-19 pneumonia’ CT images versus ‘normal’ and ‘pneumonia’. The metrics required for each class are precision, F1-score, macro-average, and weighted average. These provided valuable insights into each model’s characteristics and identified areas for improvement to make informative decisions for model refinement and optimisation.

Tables 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, and 4.9 describe the performance metrics for DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, and ResNeXt 101 and the proposed model. Table 4.10 compares the precision values of the CNN architecture models, including the proposed model. Table 4.11 compares the recall of all the CNN architectures and proposed models. Finally, Table 4.12 compares the F1-score of the existing CNN architecture models and the proposed model.

Table 4.3: DenseNet 201 Performance Metrics.

	DenseNet201			
	precision	recall	f1-score	support
Normal	0.9825	1.0000	0.9912	168
Pneumonia	1.0000	0.9703	0.9849	101
COVID-19	1.0000	1.0000	1.0000	732
accuracy	0.9970			
macro avg	0.9942	0.9901	0.9920	1001
weighted avg	0.9971	0.9970	0.9970	1001

Table 4.3 presents DenseNet 201 performance metrics for multiclass classification tasks involving ‘normal’, ‘COVID-19 pneumonia’, and ‘pneumonia’ lung CT. The metrics include precision, recall, accuracy, and F1-score for each class and the weighted and macro average values.

The DenseNet 201 model demonstrates high precision and recall for detecting ‘normal’ lung CT with values of 0.9825 and 1.0000, respectively. The F1-score is 0.9912. The model achieved a precision of 1.0000 and a recall of 0.9703 for ‘pneumonia’. The F1-score is 0.9849. The model also indicates high precision and recall for ‘COVID-19 pneumonia’ lung CT, with a precision value of 1.0000 and a recall value of 1.0000. The F1-score is 1.0000. DenseNet 201’s accuracy is 0.9970.

DenseNet 201 also demonstrates a macro-average precision value of 0.9942, a recall value of 0.9901, and an F1-score of 0.9920. The model indicates a weighted average precision, recall, and F1-score value of 0.9970.

The table above presents metrics that provide insights into the model’s performance for each class and its overall effectiveness in the multi-class classification.

Table 4.4: GoogLeNet Performance Metrics.

	GoogLeNet			
	precision	recall	f1-score	support
Normal	0.9882	1.0000	0.9941	168
Pneumonia	1.0000	0.9802	0.9900	101
COVID-19	1.0000	1.0000	1.0000	732
accuracy	0.9980			
macro avg	0.9961	0.9934	0.9947	1001
weighted avg	0.9980	0.9980	0.9980	1001

Table 4.4 presents the GoogLeNet model's high precision and recall for ‘normal’ lung CT of 0.9882 and 1.0000, respectively. It achieved an F1-score of 0.9941. The model achieved a precision of 1.0000 and a recall of 0.9802 for ‘pneumonia’. The F1-score is 0.9900. The model also indicates high precision and recall for ‘COVID-19 pneumonia’ lung CT, with a precision value of 1.0000 and a recall value of 1.0000. The F1-score is 1.0000. GoogLeNet accuracy is 0.9980.

GoogLeNet also demonstrates the macro average precision value of 0.9961, recall value of 0.9934, and F1-score of 0.9947. The model indicates a weighted average precision, recall, and F1-score value of 0.9980.

Table 4.5: ResNet 50 Performance Metrics.

	ResNet 50			
	precision	recall	f1-score	support
Normal	1.0000	1.0000	1.0000	168
Pneumonia	1.0000	1.0000	1.0000	101
COVID-19	1.0000	1.0000	1.0000	732
accuracy	1.0000			
macro avg	1.0000	1.0000	1.0000	1001
weighted avg	1.0000	1.0000	1.0000	1001

Table 4.5 indicates that the ResNet 50 model achieved 1.0000 for all the metrics in all categories.

Table 4.6: ResNet 101 Performance Metrics.

	ResNet 101			
	precision	recall	f1-score	support
Normal	0.9882	1.0000	0.9941	168
Pneumonia	1.0000	0.9802	0.9900	101
COVID-19	1.0000	1.0000	1.0000	732
accuracy	0.9980			
macro avg	0.9961	0.9934	0.9947	1001
weighted avg	0.9980	0.9980	0.9980	1001

Table 4.6 presents the ResNet 101 model's precision and recall values for 'normal' lung CT of 0.9882 and 1.0000, respectively. It obtained an F1-score of 0.9941. The model achieved a precision of 1.0000 and a recall of 0.9802 for 'pneumonia'. The F1-score is 0.9900. The model also indicates high precision and recall for 'COVID-19 pneumonia' lung CT, with a precision value of 1.0000 and a recall value of 1.0000. The F1-score is 1.0000. ResNet101's accuracy is 0.9980.

ResNet 101 also demonstrates a macro-average precision value of 0.9961, a recall value of 0.9934, and an F1-score of 0.9947. The model indicates a weighted average precision, recall, and F1-score value of 0.9980.

Table 4.7: ResNet 152 Performance Metrics.

	ResNet 152			
	precision	recall	f1-score	support
Normal	0.9941	1.0000	0.9970	168
Pneumonia	1.0000	0.9703	0.9849	101
COVID-19	0.9973	1.0000	0.9986	732
accuracy	0.9970			
macro avg	0.9971	0.9901	0.9935	1001
weighted avg	0.9970	0.9970	0.9970	1001

Table 4.7 indicates the ResNet 152 model's precision and recall values for normal lung CT are 0.9941 and 1.0000, respectively. The F1-score is 0.9970. The model achieved a

precision of 1.0000 and a recall of 0.9703 for ‘pneumonia’. The F1-score is 0.9849. The model also indicates ‘COVID-19 pneumonia’ lung CT with a precision value of 0.9973 and a recall value of 1.0000. The F1-score is 0.9986. ResNet 152’s accuracy is 0.9970.

ResNet 152 also demonstrates a macro-average precision value of 0.9971, a recall value of 0.9901, and an F1-score of 0.9935. The model indicates a weighted average precision, recall, and F1-score value of 0.9970.

Table 4.8: ResNeXt 101 Performance Metrics.

	ResNeXt101			
	precision	**recall**	**f1-score**	**support**
Normal	0.9882	1.0000	0.9941	168
Pneumonia	1.0000	0.9802	0.9900	101
COVID-19	1.0000	1.0000	1.0000	732
accuracy	0.9980			
macro avg	0.9961	0.9934	0.9947	1001
weighted avg	0.9980	0.9980	0.9980	1001

Table 4.8 demonstrates ResNeXt 101 precision and recall values for ‘normal’ lung CT of 0.9882 and 1.0000, respectively. It achieved an F1-score of 0.9941. The precision and recall values for ‘pneumonia’ lung CT are 1.0000 and 0.9802. The F1-score is 0.9900. The model also obtained 1.0000 for all precision, recall, and F1-score values for ‘COVID-19 pneumonia’ lung CT. The accuracy of the model is 0.9980. The macro average precision, recall, and F1-score values are 0.9961, 0.9934, and 0.9947. The weighted average precision, recall, and F1-score values are all 0.9980.

Table 4.9: Proposed Model Performance Metrics.

	Proposed Model			
	precision	**recall**	**f1-score**	**support**
Normal	0.9882	1.0000	0.9941	168
Pneumonia	1.0000	0.9802	0.9900	101
COVID-19	1.0000	1.0000	1.0000	732
accuracy	0.9980			
macro avg	0.9961	0.9934	0.9947	1001
weighted avg	0.9980	0.9980	0.9980	1001

Table 4.9 demonstrates the proposed model precision, recall and F1-score values for ‘normal’ lung CT of 0.9882, 1.0000, and 0.9941, respectively. It obtained 1.0000, 0.9802, and 0.9900 for ‘pneumonia’ lung CT’s precision, recall, and F1-score values. It also achieved 1.0000 for all ‘COVID-19 pneumonia’ lung CT’s precision, recall, and F1-score, and an accuracy of 0.9980. The macro average precision, recall, and F1-score are 0.9961, 0.9934, and 0.9947. The weighted average precision, recall, and F1-score values are all 0.9980.

Table 4.10: Precision in Performance Metrics.

Precision	Normal	Pneumonia	COVID-19	Acc	macro avg	weighted avg
DenseNet201	0.9825	1.0000	1.0000	0.9970	0.9942	0.9971
GoogLeNet	0.9882	1.0000	1.0000	0.9980	0.9961	0.9980
ResNet50	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
ResNet101	0.9882	1.0000	1.0000	0.9980	0.9961	0.9980
ResNet152	0.9941	1.0000	0.9973	0.9970	0.9971	0.9970
ResNeXt101	0.9882	1.0000	1.0000	0.9980	0.9961	0.9980
Proposed Model	0.9882	1.0000	1.0000	0.9980	0.9961	0.9980

Table 4.10 compares precision values between DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152 and the proposed model. ResNet 50 achieved 1.0000, the highest value in all three classes: accuracy, macro, and weighted average. GoogLeNet, ResNet 101, ResNeXt 101, and the proposed model obtained the same precision values of 0.9882, 1.0000, 1.0000, 0.9980, 0.9961, and 0.9980 for all three classes: accuracy, macro, and weighted average, respectively. ResNet152 achieved a precision value of 0.9941, 1.0000, and 0.9973 for 'normal', 'pneumonia', and 'COVID-19 pneumonia' lung CTs. It also obtained 0.9970, 0.9971, and 0.9970 for accuracy, macro, and weighted average. DenseNet201 acquired 0.9825, 1.0000, and 1.0000 for 'normal', 'pneumonia', and 'COVID-19 pneumonia' lung CTs. It also obtained 0.9970, 0.9942, and 0.9971 for accuracy, macro, and weighted average.

Precision is an essential metric in multiclass classification. It measures the architecture models' capability to predict positive instances for each class correctly. Table 4.10 shows that all of the existing architecture models and the proposed model obtained high precision, indicating that the models had a low rate of false positives for all categories of 'normal', 'pneumonia', and 'COVID-19 pneumonia' CT images. It means that when the model predicted a sample of the CT image, it correctly identified that the image belonged to a particular class. It is a desirable trait as it reflects the model's accuracy in identification. ResNet 50 showed that its precision surpassed the existing and proposed models with a value of 1.0 for identifying normal, pneumonia, and COVID-19 CT images. This was followed by ResNet 152 with a precision value of 0.9941 for 'normal' and 1.0 for 'pneumonia', and 0.9973 for 'COVID-19 pneumonia' CT images. Subsequently, GoogLeNet, ResNet 101, ResNeXt 101, and the proposed model obtained a precision value of 0.9882 for 'normal' CT images and a 1.0 for both 'pneumonia' and 'COVID-19 pneumonia' CT images. DenseNet 201 obtained the most negligible precision value of 0.9825 for 'normal' CT images and 1.0 precision value for both 'pneumonia' and 'COVID-19 pneumonia' CT images. The precision value in multiclass classification was calculated separately for each class. Hence, ResNet 50, with the highest precision value, indicated that the model performed well in distinguishing classes from other architecture models. The other existing and proposed models also performed well, obtaining high precision scores for individual classes.

Table 4.11: Recall in Performance Metrics.

Recall	Normal	Pneumonia	COVID-19	Acc	macro avg	weighted avg
DenseNet201	1.0000	0.9703	1.0000	0.9970	0.9901	0.9970
GoogLeNet	1.0000	0.9802	1.0000	0.9980	0.9934	0.9980
ResNet50	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
ResNet101	1.0000	0.9802	1.0000	0.9980	0.9934	0.9980
ResNet152	1.0000	0.9703	1.0000	0.9970	0.9901	0.9970
ResNeXt101	1.0000	0.9802	1.0000	0.9980	0.9934	0.9980
Proposed Model	1.0000	0.9802	1.0000	0.9980	0.9934	0.9980

Table 4.11 compares the recall values between DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152 and the proposed model. ResNet 50 obtained 1.0000 for all classes, accuracy, macro, and weighted average. The GoogLeNet, ResNet 101, ResNeXt 101 and the proposed model obtained performance metrics values of 1.0000, 0.9802, 1.0000, 0.9980, 0.9934, and 0.9980 for all classes, accuracy, macro, and weighted average. ResNet152 and DenseNet 201 achieved 1.0000, 0.9703, and 1.0000 for 'normal', 'pneumonia', and 'COVID-19 pneumonia' lung CTs respectively. The accuracy, macro, and weighted average values are 0.9970, 0.9901, and 0.9970.

Recall, also known as sensitivity or the actual positive rate, it measures the proportion of correctly predicted samples for a specific class out of all the samples that belong to that class. Recall provides insights into the architecture models' ability to identify positive instances correctly. ResNet 50 obtained the highest and most consistent recall value of 1.0 for all classes. All the models obtained a recall value of 1.0 for 'normal' and 'COVID-19 pneumonia' CT images, but the result varied for 'pneumonia', with 0.9802 for GoogLeNet, ResNet 101, ResNeXt 101, and the proposed model. Lastly, DenseNet 201 and ResNet 152 obtained 1.0 for 'normal' and 'COVID-19 pneumonia' CT image recall values but achieved 0.9703 for ‘pneumonia’ CT images. ResNet 50 had the best recall values in all the classes.

Table 4.12: F1-Score in Performance Metrics.

F1-Score	Normal	Pneumonia	COVID-19	Acc	macro avg	weighted avg
DenseNet201	0.9912	0.9849	1.0000	0.9970	0.9920	0.9970
GoogLeNet	0.9941	0.9900	1.0000	0.9980	0.9947	0.9980
ResNet50	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
ResNet101	0.9941	0.9900	1.0000	0.9980	0.9947	0.9980
ResNet152	0.9970	0.9849	0.9986	0.9970	0.9935	0.9970
ResNeXt101	0.9941	0.9900	1.0000	0.9980	0.9947	0.9980
Proposed Model	0.9941	0.9900	1.0000	0.9980	0.9947	0.9980

Table 4.12 compares the F1-score values between DenseNet 201, GoogLeNet, ResNet 50, ResNet 101, ResNet 152, ResNeXt 101 and the proposed model. ResNet 50 achieved 1.0000 on all classes, accuracy, macro, and weighted average. GoogLeNet, ResNet 101, ResNeXt 101 and the proposed model achieved performance metrics values of 0.9941, 0.9900, and 1.0000 for ‘normal’, ‘pneumonia’, and ‘COVID-19 pneumonia’ lung CT. It obtained 0.9980, 0.9947, and 0.9980 accuracy, macro, and weighted average, respectively. ResNet152 shows the ‘normal’, ‘pneumonia’, and ‘COVID-19 pneumonia’ lung CT achieved 0.9970, 0.9849, and 0.9986, respectively. The accuracy, macro, and weighted average are 0.9970, 0.9935, and 0.9970. DenseNet201 indicates the ‘normal’, ‘pneumonia’, and ‘COVID-19 pneumonia’ lung CT values to be 0.9912, 0.9849, and 1.0000, respectively. It also achieved an accuracy, macro, and weighted average of 0.9970, 0.9920, and 0.9970, respectively.

Table 4.12 shows the F1-score of the existing and proposed models. It combines precision and recall into a single value and balances the two metrics during classification. ResNet 50 obtained the highest F1-score value of 1.0 for all classes. This was followed by GoogLeNet, ResNet 101, ResNeXt 101, and the proposed model with the F1-Score for ‘pneumonia’ CT images of 0.9900, ‘normal’ CT images of 0.9941, and 1.0 for ‘COVID-19 pneumonia’ CT images. ResNet 152 obtained an F1-score of 0.9970 for ‘normal’ lung CT images, 0.9986 for ‘COVID-19 pneumonia’ CT images, and 0.9849 for ‘pneumonia’ CT images. DenseNet 201 achieved the lowest F1-score for ‘normal’ lung CT images with 0.9912, 0.9849 for ‘pneumonia’, and 1.0 for ‘COVID-19 pneumonia’ CT images. Overall, ResNet 50 was the model that achieved the best results. The proposed model performed reasonably well and matched evenly with ResNeXt 101, ResNet 101, and GoogLeNet.

4.1.4 Summary

The multi-class classification study consisted of many more datasets than the binary classification. Hence, data augmentation was not performed on the labelled images. The multi-class classification showed that ResNet 50 performed the best among the rest of the CNN architecture. Although the proposed model was not the best among the CNN architectures, the results were on par with ResNeXt 101, ResNet 101, and GoogLeNet. Table 4.1 describes ResNeXt 101 as having the highest training parameters, 86,748,483, indicating that the architecture requires the highest resources to run the model. The proposed model with 7,724,523 training parameters, 11 times less than ResNeXt 101, achieved similar accuracy.

4.2 Research Limitation

The research applied small datasets to train the CNN architectures for binary classification, resulting in a total of 746 images. The existing and proposed models are intricate, with small data as input data; the models will either overfit or underfit. The quantity of the dataset was further reduced when it was divided into three categories to train, validate, and test the neural networks. The problem of data scarcity is often responsible for poor performance, which results in incomplete projects.

Six different data augmentation methods were performed on the training and validation datasets to compensate for the lack of quantity of the dataset and train the neural network to be more robust and versatile in classifying 'COVID-19 pneumonia' and 'normal' CT images. However, based on the training and validation curves, the validation loss described that the neural network models picked up noises that caused overfitting. Overfitting in the validation loss indicates that more training examples are required to improve the model performance on the unseen data. The batch normalisation technique had been utilised but produced a slight regularisation effect. Therefore, Figure 4.1 shows the fluctuation of peaks in DenseNet 201. Dropout can be used to increase the regularisation effect. The research also conducted cropping and rotation of the images as data augmentation operations to create more data for the algorithm to learn from. The algorithm overcropped the images, which, combined with rotation, exacerbated the loss of essential clues in the images. Hence, the algorithm created arbitrary images that excluded important information required to learn and classified them as 'COVID-19 pneumonia' or 'normal' CT images. Cross-validation can be applied to increase the amount of training data. The learning rate of the algorithm can be lowered to improve the performance of the graph in Figure 4.1. The CNN optimisers can be improved instead of stochastic gradient descent (SGD) with momentum; ADAM can improve the algorithm.

Deep learning models require significant hardware and computational costs to train them efficiently. Although Google Colab Pro provides enhanced resources, it has limited memory and runtime duration. Complex deep-learning models or large-scale datasets exceed the available resources, requiring alternative infrastructure. Due to limitations, Google Collaboration is a free online GPU that cannot cover neural networks. It restricts users from conducting long-running tasks, especially if the models have to run with large datasets.

Unlike binary classification, multi-class classification overcame the dataset's scarcity by acquiring 10,001 CT images. Hence, data augmentation was not required to increase the images' variation. Multi-class classification experienced class imbalance, where the 'normal', 'pneumonia', and 'COVID-19' classes ratios were 2:1:7, respectively, causing it to have disproportionately small numbers for 'normal' and 'pneumonia' samples. As the number of classes increased, the complexity of the model and the computational requirements also increased. The deep learning models with more classes required extensive training and larger datasets to perform well.

The selection of appropriate evaluation metrics for multi-class classification was challenging because accuracy alone cannot provide the complete picture. Multi-class datasets might have instances with ambiguity or mislabeled samples, introducing noise and complexity in the learning process. The model's performance heavily depended on the quality and relevance of the selected features. Sometimes, certain essential features in the input may be missing, leading to suboptimal performance. CNN architectures often rely on the availability and quantity of labelled training data. The binary classification study had dataset limitations, but the data were balanced between the classes. Multi-class classification had wide ranges of datasets, but the availability of specific datasets for certain classes, like 'normal' lung CT images and 'pneumonia' lung CT images, was limited, causing the classes to be imbalanced. Hence, limited access to such resources can hinder the development and training of complex deep-learning models.

Kaggle contains a wide range of datasets, but the availability of specific datasets within the classes was limited. Certain domains or datasets categorised according to classes were unavailable for the platform. Kaggle provides pre-configured environments and restrictions on the usage of specific libraries or resources, which limits the flexibility and customization options for researchers who require specific software versions, packages, or hardware configurations.

CHAPTER 5: CONCLUSION AND FUTURE WORKS

5.1 Conclusion

The research provided a thorough understanding of the deep learning application for this multiclass classification task and the distinct opportunities and problems associated with each image class. The proposed model design prioritises deployability and lightweight characteristics, ensuring it can be easily integrated into clinical settings without imposing excessive computational or storage demands. The proposed model achieved an accuracy, F1-score, recall, and precision of 0.9980 despite not being pre-trained with the ImageNet dataset. All the objectives were achieved.

This binary classification study between 'COVID-19 pneumonia' and 'normal' CT images applied various transfer learning models and comprehensive analysis to automatically distinguish the images by machines. Firstly, the images were divided into training, validation, and testing. The images, divided into training and validation categories, were augmented to increase the datasets. The images in the testing category were not augmented. Then, the binary classification of the augmented dataset was implemented on ten different pre-trained models. The results have shown that ResNeXt 101 consists of the highest number of parameters, which is 86,746,434, achieving the highest accuracy of 98%. ResNeXt 101 demonstrated robustness, achieving high accuracy and generalisation on unseen data. GoogLeNet, on the other hand, had the most minor training parameters, with 5,601,954, but it had the lowest accuracy of 76.44%.

The multi-class classification involved the study of 'normal', 'pneumonia', and 'COVID-19 pneumonia' CT images applied to various transfer learning models. A proposed model was designed and analysed to distinguish the images automatically. Firstly, the CT images were divided into training, validation, and testing. The amount of data was sufficient, so data augmentation was unnecessary. However, this study dealt with imbalanced datasets, where some class ‘COVID-19 pneumonia’ CT images have significantly more samples than others. The study has shown that ResNet 50 achieved the best results with 100% precision, F1-score, recall, and accuracy, followed by the proposed model, GoogLeNet, ResNet 101, and ResNeXt 101. The proposed model, GoogLeNet, ResNet 101, and ResNeXt 101, had achieved a 0.9980 weighted average and a 0.9961 macro average. ResNet 152 achieved a macro average of 0.9971 and a weighted average of 0.9970. DenseNet 201 achieved the worst result, with a macro average of 0.9942 and a weighted average of 0.9971. ResNet 50 achieved 100% accuracy in multiclass classification as it was pre-trained on ImageNet. Therefore, it has the potential to achieve 100% accuracy in three (3) multiclass classifications.

The research introduced a novel framework that ResNeXt and Inception inspired to create a synergistic effect. The intention of the research is solely on the deployability of the model in the future. The proposed model is designed to be as lightweight as possible while performing at par with the best ImageNet models. The proposed model has 7,724,523 parameters, 11 times less than ResNeXt 101 while having similar accuracy. It makes the model lightweight and less intricate in its width and depth. The lightweight proposed model paves the way for deployment of the model for inference even in low-power devices such as embedded medical devices.

5.2 Limitations and Future Works

It is essential to acknowledge that deep learning is not without limitations. One of the challenges encountered during this binary classification was data scarcity and interpretability. These limitations remain an active area of research, and further advancements are needed to enhance the interpretability, efficiency, and generalisation of deep learning models using small data.

Despite the advantages of the proposed model mentioned during the discussion, the model is not yet optimised to be deployed for production. More work still needs to be done to ensure the model can be trained, make inferences faster, and specialise in the real-world use case. The proposed model should undergo training on the ImageNet platform to ensure its adaptability for deployment on chest CT images and diverse images available within the ImageNet platform. A lightweight model with fewer parameters certainly helps in this aspect. Additionally, it should be possible to fine-tune or adjust the model to get the same or better performance compared to ResNet 50 and classify CT images with more than three (3) classifications.

For future work, the proposed model required further and continuous refinement to integrate the algorithm for adaptation of multi-data modalities, such as combining chest CT images with various clinical data or laboratory results to improve diagnostic capabilities. The proposed model required further testing and adaptation to account for chest CT image acquisition protocols and equipment variations. The proposed model can then be integrated into the national electronic healthcare records, picture archiving, and communication systems. Subsequently, the proposed model is required to meet data privacy and security regulatory standards to facilitate the model's adoption in medical settings. By reducing data transfer and making the model workable in resource-constrained places, the possibility of implementing and integrating the suggested model on the edge of computing empowers healthcare systems and institutions to manage and utilise medical pictures promptly, utilising local image analysis.

Additionally, medical practitioners may successfully use and comprehend the model by designing user-friendly interfaces. The proposed model's research findings must be openly shared through science and medical journals. Partnerships with research institutions, healthcare organisations, and AI developers are essential to creating open-source resources and benchmark datasets. The proposed model can be further refined and integrate with explainable artificial intelligence (XAI) to enhance the proposed model's interpretability and transparency and ensure that the classification and detection of chest CT images can be explained to healthcare professionals and stakeholders. These programmes promote continuous progress in the field and improve our understanding of COVID-19 CT image analysis. Through the contribution of the proposed model, early diagnosis and containment of infectious diseases are facilitated by the model's implementation in public health initiatives through collaboration with governments and public health groups.

REFERENCES

Adaloglou, N. (2021, January 21). *Best deep CNN architectures and their principles: from AlexNet to EfficientNet*. Retrieved from AI SUMMER: https://theaisummer.com/cnn-architectures/

Al-Tawfiq, J. A., Zumlaa, A., & Memish, Z. A. (2014, 10). Coronaviruses: Severe Acute Respiratory Syndrome Coronavirus and Middle East Respiratory Syndrome Coronavirus in Travellers. *Wolters Kluwer Health, 27*(5), 411 - 417. 10.1097/QCO.0000000000000089

Alzubaidi, L., Zhang, J., Humaidi, A. J., Al-Dujaili, A., Duan, Y., Al-Shamma, O., . . . Farhan, L. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. *Journal of Big Data, 53*(8), 1 - 74. https://doi.org/10.1186/s40537-021-00444-8

American Library Association (ALA). (2022, February 7). *Worldometer*. Retrieved from https://www.worldometers.info/coronavirus/

Apostolopoulos, I. D., & Mpesiana, T. A. (2020). COVID-19: Automatic Detection from X-ray Images Utilizing Transfer Learning with Convolutional Neural Networks. *Physical and Engineering Sciences in Medicine, 43*, 635 - 640. https://doi.org/10.1007/s13246-020-00865-4

Ardakani, A. A., Kanafi, A. R., Acharya, U. R., Khadem, N., & Mohammadi, A. (2020). Application of deep learning technique to manage COVID-19 in routine clinical practice using CT images: Results of 10 convolutional neural networks. *Computers in Biology and Medicine, 121*, 1 - 9. https://doi.org/10.1016/j.compbiomed.2020.103795

Ayyar, T. M. (2020, Nov 6). *A practical experiment for comparing LeNet, AlexNet, VGG and ResNet models with their advantages and disadvantages.* Retrieved from Medium: https://tejasmohanayyar.medium.com/a-practical-experiment-for-comparing-lenet-alexnet-vgg-and-resnet-models-with-their-advantages-d932fb7c7d17

Bao, C., Liu, X., Zhang, H., Li, Y., & Liu, J. (2020). Coronavirus Disease 2019 (COVID-19) CT Findings: A Systematic Review and Meta-analysis. *Journal of the American College of Radiology, 17*(6), 701-709. https://doi.org/10.1016/j.jacr.2020.03.006

Bengio, Y., & Delalleau, O. (2009). Justifying and Generalizing Contrastive Divergence. *Neural Computation, 21* (6), 1601–1621. https://doi.org/10.1162/neco.2008.11-07-647

Bengio, Y., & LeCun, Y. (2007). Scaling Learning Algorithms towards AI. In L. B. Weston (Ed.), *Large-scale kernel machines* (pp. 1 - 41). MIT Press. https://doi.org/10.7551/mitpress/7496.003.0016

Brownlee, J. (2019, August 19). *A Gentle Introduction to Mini-Batch Gradient Descent and How to Configure Batch Size*. Retrieved from Machine Learning Mastery:

https://machinelearningmastery.com/gentle-introduction-mini-batch-gradient-descent-configure-batch-size/

Brownlee, J. (2019, 1 9). *A Gentle Introduction to the Rectified Linear Unit (ReLU)* . Retrieved from Machine Learning Mastery: Making Developers Awesome at Machine Learning: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

Brownlee, J. (2020, August 28). *How to Use Polynomial Feature Transforms for Machine Learning*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/polynomial-features-transforms-for-machine-learning/

Brownlee, J. (2020, August 28). *How to Use Power Transforms for Machine Learning*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/power-transforms-with-scikit-learn/

Brownlee, J. (2020, August 15). *Parametric and Nonparametric Machine Learning Algorithms*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/parametric-and-nonparametric-machine-learning-algorithms/

Brownlee, J. (2022, August 15). *When to Use MLP, CNN, and RNN Neural Networks*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/when-to-use-mlp-cnn-and-rnn-neural-networks/

Cagle Jr, S. D., Landrum, L. S., & Kennedy, A. M. (2023). Chronic Obstructive Pulmonary Disease: Diagnosis and Management. *American Family Physician, 107*(6), 604 - 612.

Carvelli, L., N.Olesen, A., AndreasBrink-Kjær, B.Leary, E., Peppard, P. E., Mignot, E., . . . Jennum, P. (2020). Design of a deep learning model for automatic scoring of periodic and non-periodic leg movements during sleep validated against multiple human experts. *Sleep Medicine, 69*, 109-119. https://doi.org/10.1016/j.sleep.2019.12.032

CDC. (2019, October 21). *Information on Rapid Molecular Assays, RT-PCR, and other Molecular Assays for Diagnosis of Influenza Virus Infection*. Retrieved from Centers for Disease Control and Prevention: https://www.cdc.gov/flu/professionals/diagnosis/molecular-assays.htm#novel

Celine, J., & Kumar, A. (2021). A Summary of Literature Review: Convolutional Neural Networks. *National Conference on Smart Systems and Technologies.* International Journal of Innovative Research in Technology. Retrieved from https://ijirt.org/master/publishedpaper/IJIRT154167_PAPER.pdf

Centers for Disease Control and Prevention. (2021, June 14). *Nucleic Acid Amplification Tests (NAATs)*. Retrieved from Centers for Disease Control and Prevention: https://www.cdc.gov/coronavirus/2019-ncov/lab/naats.html

Chen, C.-F., Fan, Q., & Panda, R. (2021). CrossViT: Cross-Attention Multi-Scale Vision Transformer for Image Classification. *CoRR, abs/2103.14899*(2103.14899), 1 - 12. https://doi.org/10.48550/arXiv.2103.14899

Chen, C.-Y., Chang, C.-k., Chang, Y.-W., Sue, S.-C., Bai, H.-I., Riang, L., . . . Huang, T.-h. (2007). Structure of the SARS Coronavirus Nucleocapsid Protein RNA-binding Dimerization Domain Suggests a Mechanism for Helical Packaging of Viral RNA. *J Mol Biol., 368*(4), 1075–1086. doi: 10.1016/j.jmb.2007.02.069

Chen, N., Zhou, M., Dong, X., Qu, J., Gong, F., Han, Y., . . . Zhang, L. (2020). Epidemiological and clinical characteristics of 99 cases of 2019 novel coronavirus pneumonia in Wuhan, China: a descriptive study. *Lancet, 395*(10223), 507 - 513. https://doi.org/10.1016/S0140-6736(20)30211-7

Cheng Jin, W. C., Xu, Z., Tan, Z., Zhang, X., Deng, L., Zheng, C., . . . Feng, J. (2020). Development and evaluation of an artificial intelligence system for COVID-19 diagnosis. *Nature Communications, 5088*(11), 1 - 14. https://doi.org/10.1038/s41467-020-18685-1

Cheng, Z., Lu, Y., Cao, Q., Qin, L., Pan, Z., Yan, F., & Yang, W. (2020). Clinical Features and Chest CT Manifestations of Coronavirus Disease 2019 (COVID-19) in a Single-Center Study in Shanghai, China. *American Journal of Roentgenology, 215*(1), 121 - 126. https://doi.org/10.2214/AJR.20.22959

Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. *Computer Vision and Pattern Recognition CoRR, abs/1610.02357*(1610.02357), 8. https://doi.org/10.48550/arXiv.1610.02357

Chouhan, V., Singh, S. K., Khamparia, A., Gupta, D., Tiwari, P., Moreira, C., . . . Albuquerque, V. H. (2019). A Novel Transfer Learning Based Approach for Pneumonia Detection in Chest X-ray Images. *Applied Science, 10*(2), 1 - 17. https://doi.org/10.3390/app10020559

Chung, M., Bernheim, A., Mei, X., Zhang, N., Huang, M., Zeng, X., . . . Shan, H. (2019). CT imaging features of 2019 novel coronavirus (2019-nCoV). *Radiology, 295*(1), 202-207. 10.1148/radiol.2020200230

Cireşan, D., Meier, U., Masci, J., & Schmidhuber, J. (2012). Multi-column deep neural network for traffic sign classification. *Neural Networks, 32*, 333-338. https://doi.org/10.1016/j.neunet.2012.02.023

Cohen, J. P., Morrison, P., & Dao, L. (2020). COVID-19 Image Data Collection. *Image and Video Processing, arXiv:2003.11597*, 1 - 4. https://doi.org/10.48550/arXiv.2003.11597

Coronavirus disease (COVID-19). (2020). Retrieved from World Health Organisation (WHO): https://www.who.int/health-topics/coronavirus#tab=tab_3

Couprie, C., Farabet, C., Najman, L., & LeCun, Y. (2013). Indoor Semantic Segmentation using depth information. *First International Conference on Learning Representations (ICLR 2013)*, (pp. 1 - 8). Scottsdale, AZ. hal-00805105

Cozzi, D., Albanesi, M., Cavigli, E., Moroni, C., Bindi, A., Luvarà, S., . . . Miele, V. (2020). Chest X-ray in new Coronavirus Disease 2019 (COVID-19) infection: findings and correlation with clinical outcome. *Radiol Med. , 125*(8), 730 - 737. 10.1007/s11547-020-01232-9

Dai, W., Zhang, H., Yu, J., Xu, H., Chen, H., Luo, S., . . . Lin, F. (2020). CT Imaging and Differential Diagnosis of COVID-19. *Can Assoc Radiol J, 71*(2), 195 - 200. doi: 10.1177/0846537120913033

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zha, X., T. U., . . . Houlsby, N. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *Computer Vision and Pattern Recognition, abs/2010.11929*(2010.11929), 1 - 22. https://doi.org/10.48550/arXiv.2010.11929

Duzgun, S. A., Durhan, G., Demirkazik, F. B., Akpinar, M. G., & Ariyurek, O. M. (2020). COVID-19 Pneumonia: The Great Radiological Mimicker. *Insights Into Imaging, 11*(118). https://doi.org/10.1186/s13244-020-00933-z

Erhan, D., Manzagol, P.-A., Bengio, Y., Bengio, S., & Vincent, P. (2009). The Difficulty of Training Deep Architectures and the Effect of Unsupervised Pre-Training. *roceedings of the Twelfth International Conference on Artificial Intelligence and Statistics* (pp. 153-160). Proceedings of Machine Learning Research. Retrieved from https://proceedings.mlr.press/v5/erhan09a.html.

Eurorad. (2020, 2 22). Retrieved from European Society of Radiology: https://www.eurorad.org/advanced-search?search=COVID&sort_by=published_at&sort_order=DESC&page=0&filter%5B0%5D=section%3A40&filter%5B1%5D=technique%3A75

Executive Committee of ICTV. (2019, July). *Taxonomy History*. Retrieved from International Committtee on Taxonomy of Viruses (ICTV): https://talk.ictvonline.org//taxonomy/p/taxonomy-history?taxnode_id=201851847

Fan, Y., Zhao, K., Shi, Z., & Zhou, P. (2019). Bat Coronaviruses in China. *Viruses, 11*(3), 210. doi: 10.3390/v11030210

Farabet, C., Couprie, C., Najman, L., & LeCun, Y. (2013). Learning Hierarchical Features for Scene Labeling. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 35*(8), 1915-1929. doi: 10.1109/TPAMI.2012.231.

Fauw, J. D., Ledsam, J. R., Romera-Pared, B., Nikolov, S., Tomasev, N., Blackwell, S., . . . Meyer, C. (2018). Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature Medicine, 24*, 1342 - 1350. https://doi.org/10.1038/s41591-018-0107-6

Ferrari, R., & Angelis, M. D. (2020). COVID-19 testing. *European Heart Journal, 41*(26), 2427 - 2428. https://doi.org/10.1093/eurheartj/ehaa483

Freeman, A., & Leigh, T. (2023, July 4). *Viral Pneumonia*. Retrieved from National Library of Medicine: https://www.ncbi.nlm.nih.gov/books/NBK513286/

Fujita, H. (2020). AI-based Computer-Aided Diagnosis (AI-CAD): The Latest Revoew to Read First. *Radiological Physics and Technology, 13*, 6 - 19. https://doi.org/10.1007/s12194-019-00552-4

Gianluca Maguolo, L. N. (2020). A Critic Evaluation of Methods for COVID-19 Automatic Detection from X-Ray Images. *Electrical Engineering and Systems Science: Image and Video Processing, 76*, 1 - 10. https://doi.org/10.1016/j.inffus.2021.04.008

Goodfellow, I. J., Courville, A., & Bengio, Y. (2013). Scaling up spike-and-slab models for unsupervised feature learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 38*(8), 1902 - 1914. doi: 10.1109/TPAMI.2012.273.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning.* Cambridge: MIT Press. Retrieved from https://mitpress.mit.edu/9780262035613/deep-learning/

Google Developers. (2023, August 23). *Introduction to Transforming Data*. Retrieved from Machine Learning: https://developers.google.com/machine-learning/data-prep/transform/introduction

Gozes, O., Frid-Adar, M., Greenspan, H., Browning, P. D., Zhang, H., Ji, W., . . . Siegel, E. (2020). Rapid AI Development Cycle for the Coronavirus (COVID-19) Pandemic: Initial Results for Automated Detection & Patient Monitoring using Deep Learning CT Image Analysis. *Image and Video Processing, abs/2003.05037*, 1 - 22. https://doi.org/10.48550/arXiv.2003.05037

Guan, W., Ni, Z., Hu, Y., Liang, W., Ou, C., He, J., . . . Li, S. (2020). Clinical Characteristics of 2019 Novel Coronavirus Infection in China. *medRxiv, 382*(18), 1708-1720. 10.1056/NEJMoa2002032

Gunraj, H., Wang, L., & Wong, A. (2020). COVIDNet-CT: A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases From Chest CT Images. *Sec. Translational Medicine, 7*, 1-11. doi.org/10.3389/fmed.2020.608525

Guo, X., Liu, D., Huang, Y., Wang, Y., Mao, J., Zhou, Y., . . . Gao, X. (2023). Revolutionizing Viral Disease Vaccination: The Promising Clinical Advancements of Non-replicating mRNA Vaccines. *Virology Journal, 20*(6). https://doi.org/10.1186/s12985-023-02023-0

Gupta, T. K., & Raza, K. (2020). Optimizing Deep Feedforward Neural Network Architecture: A Tabu Search Based Approach. *Neural Processing Letters, 51*(3), 2855 - 2870. doi:10.1007/s11063-020-10234-7

Hadjiliadis, D., Jr, P. F., Dugdale, D. C., & Conaway, B. (2022, 7 31). *Viral Pneumonia*. Retrieved from Medline Plus: https://medlineplus.gov/ency/article/000073.htm

Hani, C., Saab, I., Dangeard, S., Bennani, S., Chassagnon, G., & Revel, M. P. (2020). COVID-19 Pneumonia: A Review of Typical CT Findings and Differential Diagnosis. *Diagnostic and Interventional Imaging, 101*(5), 263 - 268. https://doi.org/10.1016/j.diii.2020.03.014

He, K., Zhang, X., Ren, S., & Sun, J. (2015). Deep Residual Learning for Image Recognition. *Computer Vision and Pattern Recognition, arXiv:1512.03385*, 12. https://doi.org/10.48550/arXiv.1512.03385

Himanshu. (2019, 1 19). *Activation Functions : Sigmoid, tanh, ReLU, Leaky ReLU, PReLU, ELU, Threshold ReLU and Softmax basics for Neural Networks and Deep Learning*. Retrieved from Medium: https://himanshuxd.medium.com/activation-functions-sigmoid-relu-leaky-relu-and-softmax-basics-for-neural-networks-and-deep-8d9c70eed91e

Hoogen, J. v., Hudson, D., Bloemheuvel, S., & Atzmueller, M. (2023). Hyperparameter Analysis of Wide-Kernel CNN Architectures in Industrial Fault Detection: An Exploratory Study. *International Journal of Data and Analytics*. https://doi.org/10.1007/s41060-023-00440-6

Hossain, M. A., & Sajib, M. S. (2019). Classification of Image Using Convolutional Neural Network. *Global Journal of Computer Science and Technology: D Neural and Artificial Intelligence, 19*(2), 1 - 7. doi:10.34257/GJCSTDVOL19IS2PG13

Hosseiny, M., Kooraki, S., Gholamrezanezhad, A., Reddy, S., & Myers, L. (2020). Radiology Perspective of Coronavirus Disease 2019 (COVID-19): Lessons From Severe Acute Respiratory Syndrome and Middle East Respiratory Syndrome. *Cardiopulmonary Imaging, 214*(5), 1078–1082. https://doi.org/10.2214/AJR.20.22969

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., . . . Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *CoRR, abs/1704.04861*(1704.04861), 1 - 9. https://doi.org/10.48550/arXiv.1704.04861

Hu, J., Shen, L., Albanie, S., Sun, G., & Wu, E. (2017). Squeeze-and-Excitation Networks. *Computer Vision and Pattern Recognition, abs/1709.01507*(1709.01507), 1 - 13. https://doi.org/10.48550/arXiv.1709.01507

Hu, X., Chu, L., Pei, J., Liu, W., & Bian, J. (2021). Model Complexity of Deep Learning: A Survey. *Knowledge and Information Systems, 63*, 2585 - 2619. https://doi.org/10.1007/s10115-021-01605-0

Huang, G., Liu, Z., Maaten, L. v., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. *Computer Vision and Pattern Recognition, arXiv:1608.06993*, 9. https://doi.org/10.48550/arXiv.1608.06993

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *Computer Vision and Pattern Recognition, arXiv:1602.07360*, 13. https://doi.org/10.48550/arXiv.1602.07360

Islam, M. R., & Matin, A. (2020). Detection of COVID-19 from CT Image by The Novel LeNet-5 CNN Architecture. *2020 23rd International Conference on Computer and Information Technology (ICCIT), 9392723*, pp. 1 - 5. DHAKA, Bangladesh. doi: 10.1109/ICCIT51783.2020.9392723.

Iyer, T. J., Raj, A. N., Ghildiyal, S., & Nersisson, R. (2021). Performance Analysis of Lightweight CNN Models to Segment Infectious Lung Tissues of COVID-19 Cases from Tomographic Images. *PeerJ. Computer Science, 7*, e368. https://doi.org/10.7717/peerj-cs.368

Jenssen, H. B. (2020, Mar 28). *COVID-19 Radiology — Data Collection and Preparation for Artificial Intelligence*. Retrieved from Medium: https://medium.com/@hbjenssen/covid-19-radiology-data-collection-and-preparation-for-artificial-intelligence-4ecece97bb5b

Jozwik, K., Kriegeskorte, N., Storrs, K., & Mur, M. (2017). Deep Convolutional Neural Networks Outperform Feature-Based But Not Categorical Models in Explaining Object Similarity Judgments. *Frontier Psychology, 8*(1726). https://doi.org/10.3389/fpsyg.2017.01726

Kahou, S. E., Pal, C., Bouthillier, X., Froumenty, P., Gülçehre, Ç., Memisevic, R., . . . Dauphin, Y. (2013). Combining Modality-specific Deep Neural Networks for Emotion Recognition in Video. *ICMI '13: Proceedings of the 15th ACM on International conference on multimodal interaction* (pp. 543–550). New York City, NY, USA: Association for Computing Machinery. https://doi.org/10.1145/2522848.2531745

Kandola, A. (2020, September 28). *What are the different types of coronavirus?* Retrieved from Medical News Today: https://www.medicalnewstoday.com/articles/types-of-coronavirus

Kanne, J. P. (2020). Chest CT Findings in 2019 Novel Coronavirus (2019-nCoV) Infections from Wuhan, China: Key Points for the Radiologist. *Radiology, 295*(1), 16 - 17. 10.1148/radiol.2020200241

Kaushik, A. (2022). *Understanding ResNet 50 Architecture*. Retrieved from Open Genus: https://iq.opengenus.org/resnet50-architecture/

Kazemnejad, A. (2019, August 30). *How to do Deep Learning research with absolutely no GPUs - Part 2*. Retrieved from Amirhossein Kazemnejad's Blog: https://kazemnejad.com/blog/how_to_do_deep_learning_research_with_absolut ely_no_gpus_part_2/

Kermany, D. S., Goldbaum, M., Cai, W., Valentim, C. C., Liang, H., Baxter, S. L., . . . Hewet, S. (2018). Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning. *Cell, 172*(5), 1122 - 1131. doi: 10.1016/j.cell.2018.02.010.

Kevin Markham. (2014, 3 25). *Simple guide to confusion matrix terminology*. (Data School) Retrieved 5 25, 2021, from https://www.dataschool.io/simple-guide-to-confusion-matrix-terminology/

Koo, H. J., Lim, S., Choe, J., Choi, S.-H., Sung, H., & Do, K.-H. (2018). Radiographic and CT Features of Viral Pneumonia. *Radiographics, 38*(3), 719-739. https://doi.org/10.1148/rg.2018170048

Koval, A., Mansouri, S. S., & Kanellakis, C. (2023). Machine learning for ARWs. In G. Nikolakopoulos, S. S. Mansouri, & C. Kanellakis, *Aerial Robotic Workers* (pp. 159-174). Butterworth-Heinemann. https://doi.org/10.1016/B978-0-12-814909-6.00016-0

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012, May 24). ImageNet Classification with Deep Convolutional Neural Networks. *NIPS'12: Proceedings of the 25th International Conference on Neural Information Processing Systems -, 1*(6), 1097–1105.

Kumar, A. (2021, November 7). *Different Types of CNN Architectures Explained: Examples*. Retrieved from Data Analytics Data, Data Science, Machine Learning, AI: https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/

Le, T. T., Cramer, J. P., Chen, R., & Mayhew, S. (2020). Evolution of the COVID-19 Vaccine Development Landscape. *Nature Reviews Drug Discovery, 19*, 667 - 668. doi: https://doi.org/10.1038/d41573-020-00151-8

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature, 521*, 436 - 444. https://doi.org/10.1038/nature14539

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-Based Learning Applied to Document Recognition. *Proceedings of the IEEE, 86*(11), 2278 - 2324. 10.1109/5.726791

Lee, H. S., Myung, J., Choi, M. J., Shin, H. J., Park, I., Chung, S. P., & Kim, J. H. (2022). Risk factors for computed tomography interpretation discrepancy in emergently transferred patients. *World Journal of Emergency Medicine, 13*(1), 54 - 58. doi: 10.5847/wjem.j.1920-8642.2022.001

Leeuwan, K. v., Rooji, M. d., Schalekamp, S., Ginneken, B. v., & Rutten, M. (2022). How Does Artificial Intelligence in Radiology Improve Efficiency and Health Outcomes? *Pediatric Radiology, 52*, 2087 - 2093. https://doi.org/10.1007/s00247-021-05114-8

Li, M. (2020). Chest CT Features and their Role in COVID-19. *Radiology Infection Disease, 7*(2), 51 - 54. doi: 10.1016/j.jrid.2020.04.001

Li, P., Fu, J. B., Li, K. F., Liu, J. N., Wang, H. L., Liu, L. J., . . . Yan, J. B. (2020). Transmission of COVID-19 in the terminal stage of incubation period: a familial cluster. *Int J Infect Disease, 96*, 452 - 453. 10.1016/j.ijid.2020.03.027

Li, Y., Gu, S., Gool, L. V., & Timofte, R. (2019). Learning Filter Basis for Convolutional Neural Network Compression. *Computer Vision and Pattern Recognition, arXiv:1908.08932*(1908.08932), 1 - 14. https://doi.org/10.48550/arXiv.1908.08932

Li, Y.-D., Chi, W.-Y., Su, J.-H., Ferrall, L., Hung, C.-F., & T-C, W. (2020). Coronavirus vaccine development: from SARS and MERS to COVID-19. *Journal of Biomedical Science, 104*(27), 1 - 23. https://doi.org/10.1186/s12929-020-00695-2

Liang, J., Ye, G., Guo, J., Huang, Q., & Zhang, S. (2021). Reducing False-Positives in Lung Nodules Detection Using Balanced Datasets. *Frontier Public Health, 9*(2021). https://doi.org/10.3389/fpubh.2021.671070

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., . . . Guo, B. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *CORR, abs/2103.14030*(2103.14030), 1 - 14. https://doi.org/10.48550/arXiv.2103.14030

Lu, R. J., Zhao, X., Li, J., Niu, P., Yang, B., Wu, H., . . . Meng, Y. (2020). Genomic characterisation and epidemiology of 2019 novel coronavirus: implications for virus origins and receptor binding. *Elsevier, 395*(10224), 565 - 574. doi: 10.1016/S0140-6736(20)30251-8

Marques, J. A., Gois, F. N., Madeiro, J. P., Li, T., & Fong, S. J. (2022). *Artificial neural network-based approaches for computer-aided disease diagnosis and treatment.* Academic Press. https://doi.org/10.1016/B978-0-323-85751-2.00008-6

Masters, D., & Luschi, C. (2018). Revisiting Small Batch Training for Deep Neural Networks. *Machine Learning, arXiv: 1804.07612*(1804.07612), 18. https://doi.org/10.48550/arXiv.1804.07612

Merkh, T., & Montufar, G. (2019). Stochastic Feedforward Neural Networks: Universal Approximation. *Machine Learning, arXiv: 1910.09763*(1910.09763), 42. https://doi.org/10.48550/arXiv.1910.09763

Moore, S. (2021, September 28). *History of COVID-19*. Retrieved from News Medical & Life Sciences: https://www.news-medical.net/health/History-of-COVID-19.aspx

Müller, D., Rey, I. S., & Kramer, F. (2020). Automated Chest CT Image Segmentation of COVID-19 Lung Infection based on 3D U-Net. *Image and Video Processing, arXiv:2007.04774*, 1 - 9. https://doi.org/10.48550/arXiv.2007.04774

Nagarajan, P., Warnell, G., & Stone, P. (2018, September 15). Deterministic Implementations for Reproducibility in Deep Reinforcement Learning. *Artificial Intelligence, arXiv:1809.05676* . https://doi.org/10.48550/arXiv.1809.05676

Nanos, G. (2023, June 19). *Deep Neural Networks: Padding*. Retrieved from Baeldung CS: https://www.baeldung.com/cs/deep-neural-networks-padding#:~:text=Causal%20Padding%3A%20This%20type%20of,values%20of%20early%20time%20steps.

Narin, A., Kaya, C., & Pamuk, Z. (2021). Automatic Detection of Coronavirus Disease (COVID-19) Using X-ray Images and Deep Convolutional Neural Networks. *Pattern Analysis and Applications, 24*, 1207–1220. https://doi.org/10.1007/s10044-021-00984-y

National Library of Medicine. (2023, February 2). *Coronavirus*. Retrieved from Medline Plus: https://medlineplus.gov/ency/article/007767.htm

Neural Magic. (2020, June 4). *How to Run ResNet at a Fraction of the Cost*. Retrieved from Neura Magic: https://neuralmagic.com/blog/sparse-resnet/

Nguyen, T. T., Nguyen, Q. V., Nguyen, D. T., Yang, S., Eklund, P. W., Huynh-The, T., . . . Hsu, E. B. (2020). Artificial Intelligence in the Battle against Coronavirus (COVID-19): A Survey and Future Research Directions. *Computer Science Computers and Society, arXiv:2008.07343*, 1 - 27. https://doi.org/10.48550/arXiv.2008.07343

Niu, R., Ye, S., Li, Y., Ma, H., Xie, X., Hu, S., . . . Chen, J. (2021). Chest CT Features Associated with the Clinical Characteristics of Patients with COVID-19 Pneumonia. *Annals of Medicine, 53*(1), 169 - 180. doi: 10.1080/07853890.2020.1851044

Nwe, M. M., & Lynn, K. T. (2019). KNN-Based Overlapping Samples Filter Approach for Classification of Imbalanced Data . *International Conference on Software Engineering Research, Management and Application, 845*, 55 - 73. https://doi.org/10.1007/978-3-030-24344-9_4

Ohata, E. F., Bezerra, G. M., Chagas, J. V., Neto, A. V., Albuquerque, A. B., & Albuquerque, V. H. (2021). Automatic Detection of COVID-19 Infection Using Chest X-Ray Images Through Transfer Learning. *IEEE/CAA Journal of Automatica Sinica, 8*(1), 239 - 248. 10.1109/JAS.2020.1003393

Padilla, T. B. (2021, February 24). *No one is safe unless everyone is safe*. Retrieved from BusinessWorld: https://www.bworldonline.com/no-one-is-safe-unless-everyone-is-safe/

Pandey, A. K. (2020, June 25). *Convolution, Padding, Stride, and Pooling in CNN*. Retrieved from Medium: https://medium.com/analytics-vidhya/convolution-padding-stride-and-pooling-in-cnn-13dc1f3ada26

Park, J. Y., Freer, R., Stevens, R., Soneji, N., & Jones, N. (2021, March 4). *The accuracy of chest CT in the diagnosis of COVID-19: An umbrella review*. Retrieved from Centre for Evidence-Based Medicine: https://www.cebm.net/covid-19/the-accuracy-of-chest-ct-in-the-diagnosis-of-covid-19-an-umbrella-review/#:~:text=Among%20symptomatic%20adult%20patients%2C%20chest,reportedly%20between%2025%20and%2083%25.

Park, M., Thwaites, R., & Openshaw, P. (2020). COVID-19: Lessons from SARS and MERS. *European Journal of Immunology, 50*(3), 308 - 311. doi: 10.1002/eji.202070035

Park, S. (2021, July 30). *Is Batch Normalization harmful? Improving Normalizer-Free ResNets*. Retrieved from Medium: https://medium.com/geekculture/is-batch-normalization-harmful-improving-normalizer-free-resnets-cf44f2fc0b2e

Perera, A. (2018, September 2). *What is Padding in Convolutional Neural Network's(CNN's) padding: (Multi-Class image classification step by step guide part 4)*. Retrieved from Medium: https://ayeshmanthaperera.medium.com/what-is-padding-in-cnns-71b21fb0dd7

PyTorch. (2017). *Illustration of Transforms*. (PyTorch) Retrieved 8 2, 2021, from https://pytorch.org/vision/stable/auto_examples/plot_transforms.html#sphx-glr-auto-examples-plot-transforms-py

Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., & Dollár, P. (2020). Designing Network Design Spaces. *Computer Vision and Pattern Recognition, abs/2003.13678*(2003.13678), 1 - 12. https://doi.org/10.48550/arXiv.2003.13678

Ritchie, H., Mathieu, E., Rodés-Guirao, L., Appel, C., Giattino, C., Ortiz-Ospina, E., . . . Beltekian, D. (2020). *Coronavirus Pandemic (COVID-19)*. Retrieved from OurWorldInData.org: https://ourworldindata.org/covid-vaccinations#citation

Rohanian, O., Kouchaki, S., Soltan, A., Yang, J., Rohanian, M., Yang, Y., & Clifton, D. (2023). Privacy-Aware Early Detection of COVID-19 Through Adversarial Training. *IEEE Journal of Biomedical and Health Informatics, 27*(3), 1249 - 1258. 10.1109/JBHI.2022.3230663

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. *CoRR, abs/1505.04597*, 1 - 8. https://doi.org/10.48550/arXiv.1505.04597

Roth, H. R., Lee, C. T., Shin, H.-C., Seff, A., Kim, L., Yao, J., . . . Summers, R. M. (2015). Anatomy-specific classification of medical images using deep convolutional nets. *Biomedical Imaging (ISBI)*, 101 - 104. https://doi.org/10.48550/arXiv.1504.04003

Rubin, G. D., Ryerson, C. J., Haramati, L. B., Sverzellati, N., Kanne, J. P., Raoof, S., . . . Bush, A. (2020). The Role of Chest Imaging in Patient Management during the COVID-19 Pandemic: A Multinational Consensus Statement from the Fleischner Society. *Radiology, 296*(1), 172 - 180. https://doi.org/10.1148/radiol.2020201365

Sahoo, S. (2018, August 20). *Deciding optimal kernel size for CNN*. Retrieved from Medium: https://towardsdatascience.com/deciding-optimal-filter-size-for-cnns-d6f7b56f9363

Salehi, S., Abedi, A., Balakrishnan, S., & Gholamrezanezhad, A. (2020). Coronavirus Disease 2019 (COVID-19): A Systematic Review of Imaging Findings in 919 Patients. *American Journal of Reoentgenology, 215*(1), 87 - 93. https://doi.org/10.2214/AJR.20.23034

Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., & Chen, L.-C. (2019). MobileNetV2: Inverted Residuals and Linear Bottlenecks. *Computer Vision and Pattern Recognition, arXiv:1801.04381*, 14. https://doi.org/10.48550/arXiv.1801.04381

Sarker, I. (2021). Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions. *SN Computer Science, 2*(420). https://doi.org/10.1007/s42979-021-00815-1

Sarvamangala, D. R., & Kulkarni, R. V. (2022). Convolutional Neural Networks in Medical Iage Understanding: A Survey. *Evolutionary Intelligence, 15*, 1 - 22. https://doi.org/10.1007/s12065-020-00540-3

Seladi-Schulman, J. (2020, October 10). *Healthline*. Retrieved from How is COVID-19 pneumonia different from regular pneumonia?: https://www.healthline.com/health/coronavirus-pneumonia#vs-regular-pneumonia

Sermanet, P., Chintala, S., & LeCun, Y. (2012). Convolutional Neural Networks Applied to House Numbers Digit Classification. *Neural and Evolutionary Computing, arXiv:1204.3968*, 1 - 4. https://doi.org/10.48550/arXiv.1204.3968

Shahid, M. (2019, Feb 25). *Convolutional Neural Network: Learn Convolutional Neural Network from basic and its implementation in Keras*. Retrieved from Medium: https://towardsdatascience.com/covolutional-neural-network-cb0883dd6529#:~:text=Stride%20denotes%20how%20many%20steps,to%20the%20input%20matrix%20symmetrically

Shukla, D. (2021, September 28). *DNA vs. mRNA vaccines: Similarities and differences*. Retrieved from Medical News Today: https://www.medicalnewstoday.com/articles/dna-vs-mrna-vaccines-similarities-and-differences

Simonyan, K., & Zisserman, A. (2015). Very Deep Convolutional Networks for Large-Scale Image Recognition. *Computer Vision and Pattern Recognition*, 14.

Simpson, S., Kay, F., Abbara, S., Bhalla, S., Chung, J., Chung, M., . . . Litt, H. (2020). Radiological Society of North America Expert Consensus Document on Reporting Chest CT Findings Related to COVID-19: Endorsed by the Scociety of Thoracic Radiology, the American College of Radiology, and RSNA. *Radiology: Cardiothoracic Imaging, 2*(2), 200152. https://doi.org/10.1148/ryct.2020200152

Solodovnikov, A., & Arkhipova, V. (2021, July 29). *Reliably beautiful: how we made a 3D model of SARS-CoV-2*. Retrieved from N+1: https://nplus1.ru/blog/2021/07/29/sars-cov-2-model

Sugimori, H. (2018). Classification of Computed Tomography Images in Different Slice Positions Using Deep Learning. *Healthcare Engineering, 2018*(1753480), 1 - 9. https://doi.org/10.1155/2018/1753480

Summer, R. M. (2017). *Deep Learning and Computer-Aided Diagnosis for Medical Image Processing: A Personal Perspective.* (L. Le, Z. Yefeng, C. Gustavo, & Y. Lin, Eds.) https://doi.org/10.1007/978-3-319-42999-1

Szegedy, C., Ioffe, S., Vanhoucke, V., & Alemi, A. (2016). Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *Computer Vision and Pattern Recognition, abs/1602.07261*(1602.07261), 1 - 12. https://doi.org/10.48550/arXiv.1602.07261

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2015). Rethinking the Inception Architecture for Computer Vision. *CoRR, abs/1512.00567*, 1 - 10. https://doi.org/10.48550/arXiv.1512.00567

Szegedy, C., Wei, L., Yangqing, J., Sermanet, P., Reed, S., Anguelov, D., . . . Rabinovich, A. (2014, Sep 17). Going Deeper with Convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), arXiv:1409.4842*(1409.4842), 1 - 12. 10.1109/CVPR.2015.7298594

Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *CoRR, abs/1905.11946*(1905.11946), 1 - 11. https://doi.org/10.48550/arXiv.1905.11946

Tartaglione, E., Barbano, C. A., Berzovini, C., Calandri, M., & Grangetto, M. (2020). Unveiling COVID-19 from Chest X-ray with deep learning: a hurdles race with small data. *International Journal of Environmental Research and Public Health., 17*(18), 6933. https://doi.org/10.3390/ijerph17186933

Tasci, E. (2020). Voting Comnbinations-Based Ensemble of Fine-Tuned Convolutional Neural Networks for Food Image Reecognition. *Multimedia Tools and Applications, 79*, 30397–30418. https://doi.org/10.1007/s11042-020-09486-1

Topol, E. J. (2019). High-performance Medicine: The Convergence of Human and Artificial Intelligence. *Nature Medicine, 25*, 44 - 56. https://doi.org/10.1038/s41591-018-0300-7

Tulin, O., Muhammed, T., Eylul Azra, Y., Ulas Baran, B., Yildirim, O., & U.Rajendra, A. (2020). Automated Detection of COVID-19 Cases Using Deep Neural Networks with X-ray Images. *Computers in Biology and Medicine, 121*(103792), 1 - 11. https://doi.org/10.1016/j.compbiomed.2020.103792

UK Research and Innovation. (2020, March 25). *Coronavirus: the science explained - UKRI*. Retrieved from What is coronavirus? The different types of coronaviruses: https://coronavirusexplained.ukri.org/en/article/cad0003/

UK Research and Innovation. (2022, March 23). *Two years on: the UK funded research that shaped the pandemic*. Retrieved from UK Research and Innovation: https://www.ukri.org/news/two-years-on-the-uk-funded-research-that-shaped-the-pandemic/

VIPER Group COVID-19 Vaccine Tracker Team. (2022, February 7). *10 Vaccines Approved for Use by WHO*. Retrieved from COVID 19 Vaccine Tracker: https://covid19.trackvaccines.org/agency/who/

VIPER Group COVID-19 Vaccine Tracker Team. (2022). *Types of Vaccines*. Retrieved from Covid-19 Vaccine Tracker: https://covid19.trackvaccines.org/types-of-vaccines/

Wang, D., Hu, B., Hu, C., Zhu, F., Liu, X., Zhang, J., . . . Peng, Z. (2020). Clinical Characteristics of 138 Hospitalized Patients With 2019 Novel Coronavirus–Infected Pneumonia in Wuhan, China. *American Medical Association, 323*(11), 1061 - 1069. doi:10.1001/jama.2020.1585

WebMD. (2020, October 10). *Coronavirus and Pneumonia*. Retrieved from WebMD LLC. All rights reserved.: https://www.webmd.com/lung/covid-and-pneumonia#1

Weerakkody, Y. (2022, April 22). *Viral Respiratory Tract Infection*. Retrieved from Radiopaedia: https://radiopaedia.org/articles/viral-respiratory-tract-infection-1

World Health Organization. (2020, 2 24). *Coronavirus disease 2019 (COVID-19) situation report* . Retrieved from World Health Organization: www.who.int/docs/default-source/coronaviruse/

Wu, H., Xu, J., Wang, J., & Long, M. (2021). Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. *CoRR, abs/2106.13008*(2106.13008), 2106.13008. https://doi.org/10.48550/arXiv.2106.13008

Xie, S., Girshick, R., Dollá, P., Tu, Z., & He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. *Computer Vision and Pattern Recognition, arXiv:1611.05431*(1611.05431), 1 - 10. https://doi.org/10.48550/arXiv.1611.05431

Yadav, S., & Jadhav, S. (2019). Deep Convolutional Neural Network Based Medical Image Classification for Disease Diagnosis. *Journal of Big Data, 6*, 113. https://doi.org/10.1186/s40537-019-0276-2

Yang, X., He, X., Zhao, J., Zhang, Y., Zhang, S., & Xie, P. (2020). COVID-CT-Dataset: A CT Scan Dataset about COVID-19. *Machine Learning, arXiv:2003.13865*, 1 - 14. https://doi.org/10.48550/arXiv.2003.13865

Yen-Chin Liu, R.-L. K.-R. (2020). COVID-19: The first documented coronavirus pandemic in history. *Biomedical Journal, 43*(4), 328 - 333. 10.1016/j.bj.2020.04.007

Zeller, M. D., & Fergus, R. (2013). Visualizing and Understanding Convolutional Networks. *arXiv, arXiv:1311.2901*, 1-11. https://doi.org/10.48550/arXiv.1311.2901

Zhang, A., Lipton, Z., Li, M., & Smola, A. (2021). Multilayer Perceptrons. In A. Zhang, Z. Lipton, M. Li, & A. Smola, *Dive Into Deep Learning* (pp. 167 - 171). Standford University.

Zhang, C., Duan, X., Wang, L., Li, Y., Yan, B., Hu, G., . . . Tong, L. (2021). Dissociable Neural Representations of Adversarially Perturbed Images in Convolutional Neural Networks and the Human Brain. *Neuroinform, 15*(677925), 1 - 13. https://doi.org/10.3389/fninf.2021.677925

Zhang, H., Wu, C., Zhang, Z., Zhu, Y., Lin, H., Zhang, Z., . . . Smola, A. (2020). ResNeSt: Split-Attention Networks. *CoRR, abs.2004.08955*(2004.08955), 1 - 12. https://doi.org/10.48550/arXiv.2004.08955

Zhang, K., Liu, X., Shen, J., Li, Z., Sang, Y., Wu, X., . . . Wang, G. (2020). Clinically Applicable AI System for Accurate Diagnosis, Quantitative Measurements, and Prognosis of COVID-19 Pneumonia Using Computed Tomography. *Cell Press, 181*(6), 1423 - 1433. 10.1016/j.cell.2020.04.045

Zu, Z. Y., Jiang, M. D., Xu, P. P., Chen, W., Ni, Q. Q., Lu, G. M., & Zhang, L. J. (2020). Coronavirus Disease 2019 (COVID-19): A Perspective from China. *Radiology, 296*(2), E15 - E25. 10.1148/radiol.2020200490.

Zuk, O. (2020). *ResNet-50: The Basics and a Quick Tutorial*. Retrieved from datagen: https://datagen.tech/guides/computer-vision/resnet-50/