Evaluating Deep Learning for Enhanced Breast Cancer Diagnosis: A Comparative Analysis of CNN Architectures

Solyle Frankle, MSc (Med) Bioinformatics

Supervisor: Dr Musalula Sinkala

March 2025

Division of Computational Biology

University of Cape Town, Faculty of Health Science

Thesis presented for the degree of Master of Medical Science in the Department of Computational Biology, University of Cape Town, March 2025, Musalula Sinkala.

Declaration

I, Solyle Emily Frankle, hereby declare that this thesis is my own original work (except where acknowledgment to others has been indicated using the APA referencing convention) and that no part of this thesis is being or has been submitted for any other degree at The University of Cape Town or any other university.

I empower The University of Cape Town to reproduce either the whole or parts of this thesis in any manner for research purposes.

Signature Signed by candidate

Date: 24 October 2024

Acknowledgements

I would like to express my deepest gratitude to those who have helped bring this dissertation to life:

To my supervisor Musalula Sinkala:

Thank you for your continued support in my research endeavours; for always offering a helping hand, and for sharing your expertise with me during this time. I have learnt a tremendous number of lessons under your supervision, and I could not be more grateful.

To my family and friends:

I extend my deepest appreciation to you. Thank you for your financial and emotional support that helped me get me through this postgraduate. Your never-ending encouragement has been a driving force, and I am eternally grateful. I am grateful for all the laughter and love you have granted me this year. Thank you for your unwavering friendship during the stressful periods, and for offering a shoulder to cry on when it was needed.

To the JW Jagger Scholarship Fund:

My sincerest thanks are extended to you. Thank you for granting me the privilege to pursue my passion this year through your financial support. I am indebted to you for making this dream a reality.

This dissertation would have been impossible without the collective support of these people- their contributions to this have been indispensable. Thank you all for sharing this journey with me.

Table of Contents

List of Figures

List of Tables

List of Abbrevations

Abbreviation	Definition
CNN	Convolutional Neural Network
CBIO	Computational Biology
APA	American Psychological Association
JW	JW Jagger Scholarship Fund
AI	Artificial Intelligence
ML	Machine Learning
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
DL	Deep Learning
NLP	Natural Language Processing
CV	Computer Vision
DNA	Deoxyribonucleic Acid
IDC	Invasive Ductal Carcinoma
SVM	Support Vector Machine
RF	Random Forest
ROI	Region of Interest
MRI	Magnetic Resonance Imaging
ABUS	Automated Breast Ultrasound
PCA	Principal Component Analysis
HOG	Histogram of Oriented Gradients
NAS	Neural Architecture Search
EOSA	Offspring Since Cosine Algorithm
FDA	Food and Drug Administration
UK	United Kingdom
NHS	National Health Services
COVID	Coronavirus Disease
CPU	Central Processing Unit
TCGA	The Cancer Genome Atlas

Abstract

Artificial Intelligence (AI), particularly its machine learning (ML) subfield, has revolutionised various sectors, including healthcare. In breast cancer care, AI's ability to analyse vast datasets and extract complex patterns from medical images has the potential to transform diagnostics and treatment strategies. Breast cancer remains one of the most prevalent cancers affecting women globally, with early and accurate diagnosis being crucial for effective treatment. AI, through its advanced image analysis capabilities, significantly improves the accuracy and efficiency of breast cancer diagnosis, specifically in distinguishing between cancer subtypes.

Here, we aim to comparatively evaluate the performance of a series of Convolutional Neural Networks (CNNs) specifically used in breast cancer subtype classification of histology images. A custom CNN model was developed. Its performance and accuracy were evaluated against the well-established ResNet50 and EfficientNetB0 models, in predicting benign and malignant breast cancer subtypes. The results demonstrated that while the custom CNN achieved an accuracy of 65% for malignant and 67% for benign subtypes with ROC-AUC scores of 0.86 and 0.90, respectively, ResNet50 significantly outperformed both the custom model and EfficientNetB0. ResNet50 attained an accuracy of 77% in classifying malignant subtypes and 77% for benign subtypes, accompanied by ROC-AUC scores of 0.92 and 0.96, respectively. Additionally, ResNet50 exhibited higher precision (0.68 for malignant, 0.67 for benign), recall (0.65 for malignant, 0.67 for benign), and F1 scores (0.65 for malignant, 0.67 for benign) across most subtypes, underscoring its robust performance and reliability in clinical settings.

In conclusion, AI, specifically through advanced CNN architectures, can greatly enhance breast cancer diagnosis by providing more accurate subtype classifications. Future work should focus on integrating these models into clinical workflows, enabling faster and more personalised treatment planning. Moreover, continued refinement of these models, including addressing the complexities of tumour heterogeneity and incorporating multimodal data, will be crucial for their widespread adoption in oncology.

1 Introduction

Since its emergence in the 1950s, artificial intelligence (AI) has shown significant success in various domains by offering innovative solutions to challenging tasks. Machine learning (ML), a branch of AI, has shown particular success in improving efficiency across a diverse range of applications. Integrating AI technologies in medicine has significantly impacted medical research and practice[1], from facilitating advanced scientific discovery to streamlining laborious tasks. Subfields of machine learning include deep learning (DL), natural language processing (NLP), and computer vision (CV), all of which collectively contribute to the development of sophisticated models that have profound implications for healthcare[2,3]. Today, these models have extensive applications in various medical domains, ranging from diagnostics to treatment planning and patient care[4]. During the initial stages of AI discovery and research, the primary focus was on understanding the underlying technology, whereas today, efforts have shifted to focus on how best to leverage the potential this technology possesses to develop lifesaving applications[5,6]. This shift emphasises the power of leveraging advancements in AI to address critical healthcare challenges. One such promising application may be harnessed in medical imaging – where ML algorithms can be applied to histology analysis to subsequently predict cancer subtypes at a high level of accuracy.

From a biological perspective, breast cancer results from the uncontrolled proliferation of abnormal cells invading surrounding breast tissue. The cells acquire mutations in genes such as BRCA1 and BRCA2 which are vitally responsible for cell survival processes such as cell cycle regulation, DNA repair and apoptosis[7, 8]. These mutations lead to subsequent uncontrolled cell growth which may interact with existing blood vessels, and cells to form the tumour microenvironment which contributes to cancer progression. Due to its high prevalence, breast cancer is a significant global health concern and whilst early diagnosis is critical for treatment this often proves challenging since initial symptoms are subtle rendering regular screening essential[9, 10, 11]. Furthermore, breast cancer treatment is relatively complex, typically involving a combination of therapies like chemotherapy, radiation and surgery all of which are tailored to the specific patient and their cancer subtype and stage[12]. It is thus a combination of these factors that contribute to the economic

and emotional impact that is felt by patients and their families as well as healthcare systems globally.

In this dissertation, I explore the application of AI/ML to histology images for precise breast cancer subtype prediction. By integrating cutting-edge AI technologies into clinical practice, this research aims to automate a medical imaging-based diagnostic protocol for breast cancer, to reduce diagnosis time and improve treatment outcomes.

Before examining this specific application of ML to breast cancer prediction, it is critical first to understand the complexity of breast cancer itself, as well as the basic foundational principles of ML. Furthermore, analysis of current research trends in breast cancer and ML research is essential to determine the optimal strategies for using these advancements in preventative oncology. Thus, this dissertation intends to shed light on the essential research components in both breast cancer and machine learning research and how the two can be leveraged to optimise the medical imaging workflow.

1.1 Rationale of the Study

Breast cancer remains a significant global health issue, with early and accurate diagnosis being critical for successful treatment outcomes[13]. Traditional diagnostic methods, such as manual histology analysis, are not only time-consuming but also subject to variability and human error[14]. This study aims to develop and compare the performance of multiple CNN models for classifying breast cancer into malignant and benign subtypes using histology images. By exploring different CNN architectures, this research seeks to determine the most effective model for accurate classification, providing insights into their strengths and limitations. The comparative aspect will highlight the best-performing model for potential integration into clinical settings, ultimately aiming to improve diagnostic accuracy and efficiency.

1.2 Aims and Objectives

1.2.1 Aim

To develop and compare multiple CNN models for accurately classifying malignant and benign breast cancer tumours from histology images, with the objective of identifying the most efficient and effective model for clinical application.

1.2.2 Specific Objectives

1. Train multiple CNN architectures (e.g., ResNet, EfficientNet, and a custom CNN) using a comprehensive dataset of histology images to classify tumours as malignant or benign.
2. Evaluate and compare the models' performance based on metrics such as accuracy, precision, recall, F1 score, and AUC.
3. Fine-tune the hyperparameters of each model to optimize performance and generalizability.
4. Identify the best-performing model through a thorough comparative analysis and provide recommendations for further improvements or clinical applications.

2 Literature Review

2.1 Breast Cancer Overview

Breast cancer, initially documented around 460 BCE, refers to the uncontrolled proliferation of malignant cells within mammary epithelial tissue[15]. Carcinogenesis in breast cancer involves several key processes, including the evasion of programmed cell death (apoptosis), uncontrolled cell division leading to limitless growth, heightened formation of new blood vessels (angiogenesis), resistance to signals that inhibit growth, and the ability to generate signals promoting its growth, along with the potential to metastasise[16]. A combination of genetic predisposition and environmental factors influences these processes.

Breast cancer can be categorised based on its histological characteristics, with at least 18 distinct types[17]. Among these, the most prevalent is invasive breast cancer of no special type (NST), previously termed invasive ductal carcinoma (IDC). This breast cancer subtype lacks discernible growth patterns or cytological features that would otherwise categorise it into a specific subtype[18]. Additional examples of histological breast cancer subtypes include invasive lobular carcinoma, tubular carcinoma, and neuroendocrine carcinoma, each exhibiting unique characteristics that impact treatment strategies and prognostic outcomes[19]. Addressing the clinical intricacies of breast cancer also involves addressing tumour heterogeneity, which is a pivotal factor contributing to breast cancer complexity. Populations of large and diverse cancer cells within tumours significantly complicate treatment approaches, since there is rarely a universally effective solution capable of targeting all cell types simultaneously[20]. As such, tumour heterogeneity must be carefully considered when discussing treatment options among clinicians and patients.

Breast cancer continues to be the leading cancer among women, with millions being affected every year[21]. Worryingly, this number continues to rise, with more than 2.3 million women being diagnosed with the disease[22]. In recent decades, there has been a noticeable rise in both the incidence and mortality rates of breast cancer. Data from 1990 to 2016 reveals a more than double increase in breast cancer cases in 60 out of 102 countries, including Afghanistan, the Philippines, Brazil, and

Argentina[13]. Similarly, deaths due to breast cancer have doubled in 43 out of 102 countries, including Yemen, Paraguay, Libya, and Saudi Arabia[23]. Projections suggest that by 2040, the global annual incidence of new breast cancer cases could reach over 3 million, with approximately 1 million deaths[24]. This trend is particularly concerning in low- and middle-income countries, where factors such as the adoption of modern lifestyle practices (like delayed pregnancies, reduced breastfeeding, early onset of menstruation, sedentary habits, and poor dietary choices), and enhanced detection methods are expected to contribute to a further increase in breast cancer incidence[25].

Notably, breast cancer-related mortality rates are higher in transitioning countries compared to those that have already undergone socioeconomic transitions. Factors such as adopting preventive behaviours, implementing effective screening programs, and ensuring early access to treatment play pivotal roles in reducing the incidence of breast cancer and improving survival rates, particularly in low- and middle-income countries[26]. It is worth noting that while a majority of women diagnosed with breast cancer in high-income countries survive, the opposite is true for women in many low- and middle-income nations[27]. Thus, greater focus should be placed on implementing these preventative strategies in transitioning countries.

Since its first identification, the diagnosis and treatment of breast cancer have significantly improved. This comes from the advent of novel scientific discoveries that have aided in timely diagnosis and treatment. These include x-rays (which paved the way for the mammogram), radiation therapy, chemotherapy, hormone therapy (e.g. tamoxifen), genetics testing, gene expression profiling for breast cancer subtype identification, and of course, advancements in the surgical procedures which have led to what is known as the mastectomy today[28]. Over time, cultural and sexual associations linked to the breast have contributed to the stigmatisation of discussing this disease openly, confining its discourse primarily to clinical literature and educational materials until recent years[11]. This persistent stigma poses significant barriers to effective prevention, early detection, and treatment, as patients often feel hesitant to openly address the disease or its symptoms[11, 29].

Technological improvements which have enhanced the diagnosis and treatment of breast cancer, alongside active efforts to reduce the stigma behind the disease, have

contributed to improved breast cancer care. As a result, patients have been able to live longer and have a better quality of life, despite higher incidence rates, and researchers have been able to better understand the complexity behind the disease[30].

2.2 Advancing Breast Cancer Care Through Multimodal Data

Breast cancer is an incredibly complex and multifaceted disease. To help reduce this complexity, specialists categorise the disease into distinct subtypes (as described in the "*Breast Cancer Overview*" section). This assists in making both diagnosis and processing patient treatment methodologies significantly easier for patients and clinicians alike. Subtyping of breast cancer has led to several advancements in breast oncology. Particularly the introduction of personalised targeted therapy as treatment for the disease[31]. The American Cancer Society describes targeted therapy as "a type of cancer treatment that uses other substances to precisely identify and attack certain types of cancer cells"[32]. This is used to treat different breast cancer subtypes differently – by precisely triggering molecular targets within cancer cells that are involved in cancer growth, progression and spread, the cancer cell growth can be molecularly blocked, and cancer cell death may be induced[33],[34]. This precision medicine approach to breast cancer treatment has shown great success, especially when used in combination with existing methods like chemotherapy[35]. Beyond increased efficacy, specialised treatment of breast cancer has enabled a greater evolution in breast cancer therapy, with this providing a wholistically better approach to combatting the disease, than was previously possible with less specific, generalised treatment[36]. Nevertheless, breast cancer subtyping does not eradicate this complexity and the problem remains nuanced. Thus, employing ML methods becomes crucial in tackling the disease as they enable the efficient analysis of complex and diverse data types, such as histology, genomics, and clinical information, ultimately leading to more accurate diagnoses and personalised treatment strategies.

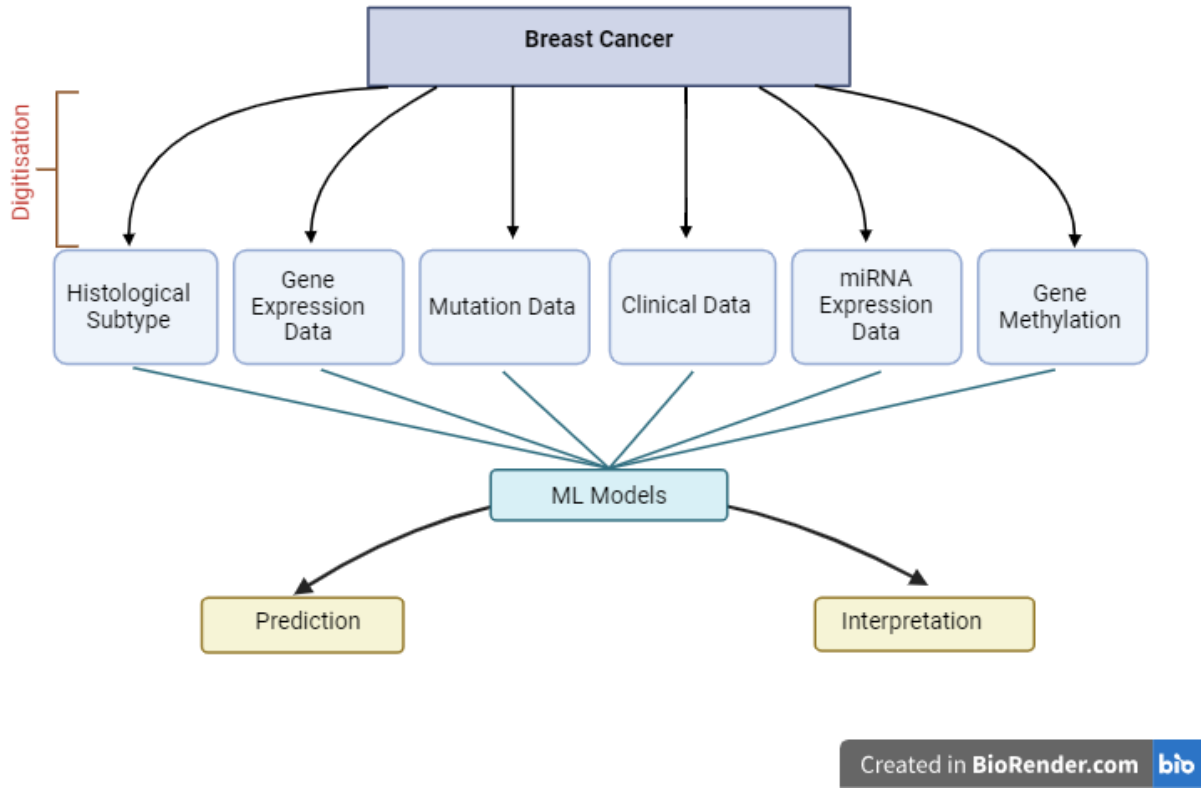


Figure 1: Using machine learning to address the multimodal components that constitute Breast Cancer

Research efforts into precision medicine techniques have drastically increased since its inclusion into everyday oncology. The natural progression of pharmacological research is towards a pharmacogenomic approach – assigning treatment therapy based on a combination of an individual's ability to absorb, distribute, and metabolise certain drugs, as well as genetic and epigenetic factors[28]. This further emphasises the benefit of subtyping breast cancer before applying targeted therapy and acts as a filter to help narrow down effective treatment options, before progressing to specialised intervention.

At present, these breast cancer subtypes are identified using immunohistochemical staining in histology images. Whilst this is the gold-standard procedure for accurate subtyping, it is flawed in the following areas:

Firstly, there is an opportunity for a large range of inter- and intra-observer variability. Different specialists may interpret the images differently, leading to inconsistencies in subtyping. Above this, the same pathologist may often interpret the same slide

differently if asked to do so at a different time of day, subsequently leading to intra-observer variability[37].

The heterogeneous nature of tumour tissue renders differentiating parts of the same tumour challenging since one consistent tumour may present varying features associated with more than one subtype. This makes deciding what subtype group the tumour belongs to more difficult, since even experienced pathologists may struggle to discern between subtypes[38].

Histology images are merely one type of data source that may be used in breast cancer analysis. Other examples of “good data” include gene expression, miRNA expression, gene methylation and clinical data, which may offer further insight into genetic or environmental factors associated with or contributing to disease progression[39, 40, 41].

Although categorising breast cancer into subtypes may seem straightforward, the actual complexity of the disease far exceeds this simplified approach. Breast cancer is multifaceted, and subtype is merely one factor of many that need to be considered when dissecting the epidemiology of the disease. Other aspects of the disease that need to be considered include gene expression data, mutation data, clinical data, miRNA expression and gene methylation. Due to the complexity of the disease, it begs to reason that imploring machine learning techniques is the most logical approach for tackling breast cancer. Machine learning can easily and effectively process such multimodal data to apply analytical techniques, make inferences, and even make informed decisions across a multidimensional space[42].

Sun et al.’s multimodal deep neural network for breast cancer prognosis prediction is a fine example[43]. By focusing on multidimensional data, Sun et al. produce a significantly more comprehensive analysis of the characteristics of breast cancer than would otherwise be achieved from a single-dimensional data study. Their model incorporates gene expression profile, copy number alteration profile and clinical data to produce a model that supersedes existing approaches like support vector machine (SVM), random forest (RF), and logistic regression in predicting breast cancer prognosis[36]. Above this, the speed and efficiency of their model are far greater than that in current medical practice. Breast cancer diagnostic units frequently experience

delays due to a shortage of specialists available to review and interpret patient scans. As a result, patients face extended waiting times for their results, as specialists must prioritise urgent, life-threatening cases daily. Meanwhile, the increasing incidence of cancer adds additional strain to the system[44].

Breakthroughs in high-throughput technology have significantly augmented the acquisition of large quantities of omics data. As a result, big data is more readily available, making data analysis significantly easier[45]. This, combined with an increased demand for AI, renders the need to incorporate multimodal data into research even more critical[46].

2.3 Machine Learning in Breast Cancer

Current histology practice requires expert pathologist opinion; however, this undeniably has various flaws, since individual judgement can be clouded by bias, rendering the process incredibly subjective. Compared with manual detection methods like histology, AI/ML-aided diagnosis can obtain similar better results and be more efficient [47, 48]. ML offers a promising, more objective future for histopathology[49]. As a primary goal, ML can be leveraged to automate detection and diagnostic pipelines in medical image analysis. Secondly, it can be used to optimise the workflow used by histopathologists and further streamline certain processes. The following section discusses an important DL technique commonly used to achieve this- the CNN.

At its most basic form, the CNN is a stacked DL algorithm that comprises several well-developed layers. These layers typically consist of an input layer, a convolutional layer, followed by a pooling layer and several fully connected layers[50]. The input layer acts as a store for the image dataset by holding pixel values[51]. The convolutional layer performs feature extraction by applying convolutional operations to the input layer (in this case, histology images) to produce a feature map. Hereafter, an activation function is applied to the resulting feature map – these may vary depending on the specific parameters of the neural network and the nuances of the problem at hand[52]. One of the most reliable and effective activations is the Rectified Linear Unit (ReLU). ReLU contributes to increased performance in many

cases of machine learning in medicine[53],[54]. Other activation functions, such as Sigmoid Activation and Hyperbolic Tangent, have also been successful. This is mainly because the Sigmoid activation function is generally applied to binary image classification problems, where the model output is a probability of an image belonging to a particular class, which can subsequently be used for prediction[55]. Additional benefits of the Sigmoid Activation function include being highly differentiable, monotonic and non-linear, which renders it ideal for gradient optimisation techniques such as backpropagation[56] , which are commonplace in image classification problems. Similarly, the Hyperbolic Tangent function is most often used for hidden neural network layers.

Transitioning to the pooling layer, this is primarily used as a down-sampling operation to reduce the spatial dimensions of the input, effectively reducing the number of parameters within the working activation[57]. This is achieved by selecting maximum values in local regions – a common optimisation technique simplifying the network whilst extracting important features. The output from the pooling layer is transformed to a 1-dimensional format through the flattening process[58]. This is essential since the fully connected layer can only process vector inputs. The fully connected layers perform functions traditionally found in Artificial Neural Networks (ANNs); this is used to generate class scores based on their activations, which are then used for classification purposes[59]. The structure of the fully connected layers is such that every neuron from one layer is connected to every other neuron in another layer.

As a result of this simple yet effective structure, CNNs are frequently used for processing and segmenting images[60, 61]. CNNs can automatically learn hierarchical features directly from raw image data. This renders CNNs highly effective at capturing patterns and spatial relationships, and, subsequently, excellent at feature extraction[62]. Furthermore, CNNs are capable of processing large datasets and leveraging these for improved performance. This, accompanied by their excellent accuracy at image classification problems, makes them the natural choice for histology image classification[63]. To reiterate, CNNs are especially good at classifying images in large datasets with potentially complex patterns. As a result, they are often the preferred choice due to their ability to learn hierarchical features from raw image data.

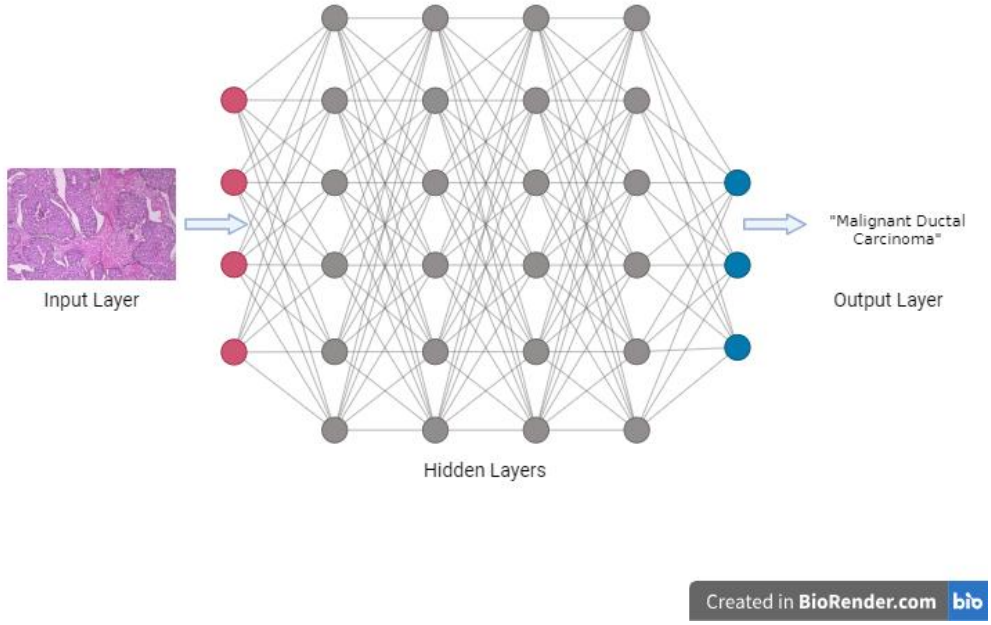


Figure 2: An overview of a basic CNN structure at work to predict breast cancer subtype from a histology image.

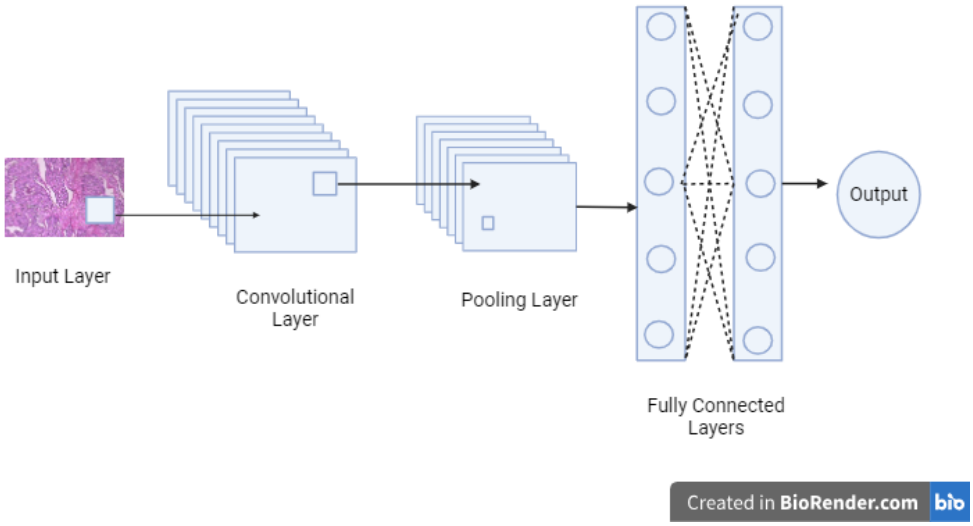


Figure 3: A more detailed overview of the hidden layers of a CNN- specifically illustrating the convolutional layer, pooling layer and fully connected layers that enable the CNN to classify its input.

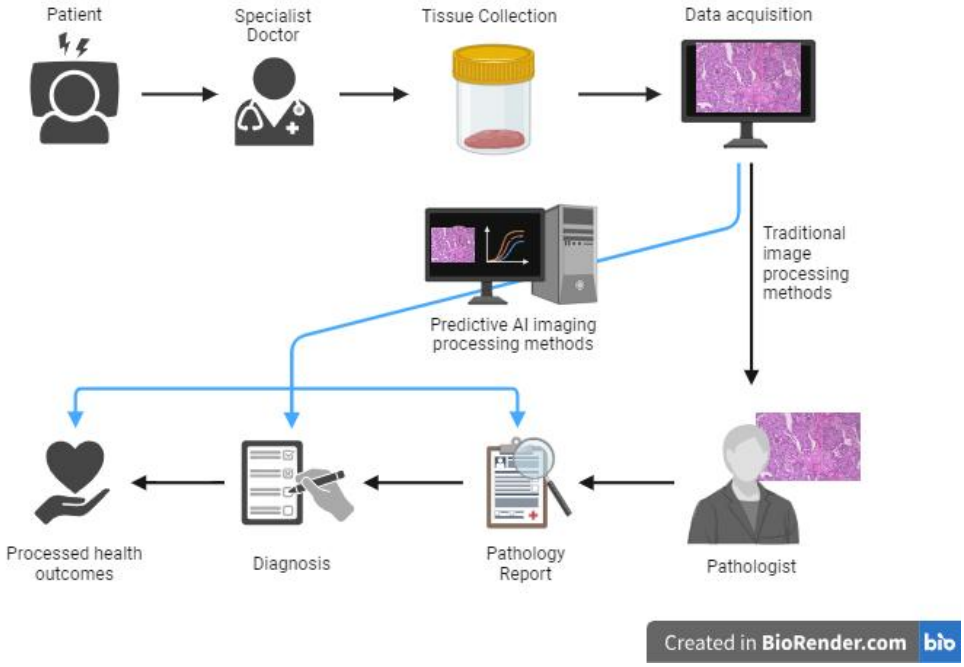


Figure 4: Traditional breast cancer diagnostic workflow, and how predictive AI can be leveraged to streamline this process.

Integrating ML technology into clinical practice has significantly benefited oncologists over the years. Notably, AI assistance has played a pivotal role, leading to a reduction in mastectomies, which were previously prone to unnecessary surgeries due to the misclassification of high-risk patient tissue biopsies[64]. This progress highlights the urgent need for ML models capable of accurately predicting high-risk cancer lesions using image-guided biopsies and updated pathological assessments[65]. Such ML-focused initiatives are paramount in minimising unwarranted surgical interventions.

Applying ML to mammogram classification has proven further success in the clinical domain. In 2019, Shen, L., Margolies, L.R., Rothstein, J.H. et al were able to produce a model which, upon exposure to a test image set, achieved a per-image AUC of 0.95 with a sensitivity of 86.7% [66]. Recently, Kebede, S.R., Waldamichael, F.G., Debelee, T.G. et al. used a dual DL model approach for mammogram assessment. They achieved an overall F-1 score of 0.87 with a sensitivity of 82%[67].

Alloqmani et al.'s model for classifying mammograms as abnormal or normal (binary classification) whilst comparing their work to other recent and relevant frameworks has also proven successful[68]. Notably, these studies extensively use the pre-processing steps to ensure quality images with isolated regions of interest (ROI) are produced before exposure to their unique ML models.

The findings from these studies show the multitude of ways that DL methods can be applied in mammography and other similar imaging platforms to achieve high levels of accuracy in classification problems. Such initiatives are vital in improving existing clinical tools to reduce false positive and false negative screening results[69].

Beyond clinical applications, ML's impact extends into cancer research, particularly in predicting survival rates and long-term cognitive outcomes. For example, in a study involving 335 high-risk cancer patients, a random forest ML model demonstrated its potential to prevent nearly one-third of unnecessary surgeries[70], showcasing the tangible benefits of ML in treatment optimisation. Given the global prominence of breast cancer among women, recent studies have increasingly harnessed ML's capabilities. These studies leverage ML models to detect visual cancer signatures and identify new prognostic factors accurately. Techniques such as neural networks, extreme gradient boosting, decision trees, and support vector machines have been pivotal in conducting precise survival analyses[71],[72],[73]. This collective evidence further emphasises ML's versatile potential in revolutionising breast cancer management across clinical and research domains.

2.4 ML Applications in Breast Cancer Subtype Differentiation

Current literature explores applications of ML techniques to breast cancer data – i.e. histology, mammography, MRI data, and among others. Of these, the most common conclusion is that machine learning significantly benefits the pathology community-increasing efficiency, correct diagnosis, and timeliness of end-to-end diagnosis. Most notably, Wang et al., 2020[74] proposed a CNN that could successfully differentiate benign and malignant breast cancer tumours in automated breast ultrasound imaging (ABUS). It was found that the model could drastically outperform Principal Component Analysis (PCA) and Histogram of Oriented Gradients (HOG) in

classification and could significantly assist human reviewers in improving their diagnosis, showing promise for the model to be incorporated into medical pathology by acting as a second reviewer within the diagnostic pipeline, subsequently increasing diagnostic reliability. Whilst this shows great promise for incorporating machine learning into existing clinical practice, there are some flaws. Namely, this method does not decrease the timeliness of diagnosis – surprisingly, diagnosis times are likely to increase, since experts will rely on the neural network as a secondary tool before finalising prognosis[65]. Secondly, the model does not use histology images, which are a significantly more accurate representation of cancer cells that may be present in breast tissue. This significantly complicates the matter since there is decreased certainty of whether the ABUS is indeed capturing true cancerous breast tissue or not. Finally, the model acts on a binary problem – classifying cancer as benign or malignant. This is a limitation since breast cancer prognosis is heavily reliant on subtyping for the subsequent application of appropriate treatment.

A novel Neural Architecture Search (NAS) based CNN model for breast cancer detection using histology images was proposed in 2021[75]. Notably, this study's hybrid NAS achieved significant success, demonstrating substantial benefits in identifying abnormalities within the histology image dataset compared to traditional manual CNN methods. Beyond the NAS, this study deployed an Offspring Since Cosine Algorithm (EOSA) to further optimise the search strategy of the used NAS. This proved to be successful in improving the search power of the CNN. While extensive research has proven the analytical success of DL in image classification problems, this study is an example of the improved performance of this already existing approach in detection and classification. By combining EOSA with NAS, a novel new type of CNN was designed, which yielded extraordinary results that far exceeded existing CNN models across several metrics: namely accuracy, sensitivity, specificity, precision, and recall. Notably, this hybrid model significantly reduced false-positive rates and classification errors. The study's proposed CNN model remains a promising contender for subsequent research endeavours. It gives researchers valuable insights into constructing networks for everyday use in digital histopathology image analysis. This study outlines several future research recommendations- exploiting swarm-based optimisation algorithms for NAS-based models or using comparative exploratory biology approaches. The overarching

recommendation, however, is to apply EOSA-NAS models to improve the present Generative Adversarial Networks (GANs) for configuring histology images. This study sheds light on the power of NAS application in CNN architecture building and how EOSA can be incorporated as an optimisation tool. This research is at the forefront of breast cancer detection in histology image analysis and urges fellow scientists to implore DL techniques to optimise the cancer prognostic pipeline.

2.5 Impact of AI/ML in Clinical Practice

Technology continues to improve in the machine learning space. More notable, however, is the rapid incorporation of machine learning technologies in the medical field. Between August 1, 2022, and July 30, 2023, the FDA approved 171 AI and machine learning devices[76]. Similarly, there has been increased demand for machine learning technologies in the UK National Health Service (NHS) to assist departments such as Service Efficiency, Diagnostic Support, Precision Medicine and Image Analysis[77]. Furthermore, there has been a significant increase in AI governance activities globally, which emphasises that there is indeed a growing need for policy controlling AI technologies, as a result of the growing presence of AI in medicine today.

Whilst research in the AI governance space extends globally, Africa is often largely excluded[78]. Nevertheless, Townsend et al. 2023 reveal that, in an African context, we are beginning to observe a similar, although sporadic, inclusion of AI in healthcare[79]. In 2020, South Africa incorporated an AI-powered X-ray system to aid in diagnosing COVID-19 during the COVID-19 pandemic[80]. In Rwanda and Tanzania, AI-driven triage systems and digital prescription delivery in mobile applications are also commonly observed[81, 82]. In Nigeria, machine learning applications to improve birth asphyxia have also been piloted[83]. These are just some examples of medical applications of AI in Africa today.

2.6 Future Direction

These clinical applications have the power to drastically transform the healthcare industry. If used correctly, AI could become a key tool for improving health equity worldwide[84]. The development of ML models and computational image analysis tools has revealed the potential to significantly improve the current mechanisms used in medical histology today[85]. ML shows a further promising impact in medical imaging and oncology. Traditionally, histology analysis is heavily reliant on specialist human expertise. While these specialists are vital in providing cancer diagnostic services, their judgments are often prone to error and internal bias and can be incredibly time-intensive. This, however, is significantly reduced in DL-based analytical approaches- the proven success of ML in medical histopathology speaks to this. Machine learning algorithms can extract more detail from images that would otherwise be overlooked by the pathologists' eye due to their high sensitivity and specificity[86]. As a result, future work may look into harnessing machine learning to reduce pathologist workloads for higher efficiency and quicker turnover times for patients suffering from breast cancer. Finally, machine learning opens the door for advanced tailored care for breast cancer patients. Precision medicine can be applied to these processes, and the analytical power of AI, and its efficient automation further harnessed to provide next-level treatment.

Whilst the future of machine learning in breast cancer care is bright, it faces several hurdles that must be overcome before its implementation in everyday practice. High validation and accuracy of these models would need to be achieved through continued research and close monitoring of their performance. Beyond this, data standardisation, regulatory compliance, and an abundance of ethical components also need to be considered by all individuals who would likely be affected by the incorporation of these machine learning technologies into healthcare to ensure ethical and safe usage.

2.7 Conclusion

To conclude, advancements in AI/ML have revolutionised various domains, with notable impacts in the medical field. Throughout this review, we have dissected the

intricacies of breast cancer, the foundational principles of machine learning, and current research trends in both fields. This study aims to optimise the medical imaging workflow for enhanced breast cancer diagnosis and management by bridging these areas.

Moving forward, the insights gained from this review catalyse further exploration and innovation in leveraging AI for preventative oncology. As technology advances, there is a growing opportunity to harness AI's capabilities for early detection, personalised treatment planning, and improved patient outcomes in breast cancer care.

This literature review contributes to the ongoing dialogue surrounding integrating AI technologies in healthcare, highlighting the potential for transformative impact on cancer diagnosis and treatment. It emphasises the importance of interdisciplinary collaboration and continuous research to unlock the full potential of AI in addressing critical healthcare challenges.

3 Methods

3.1 The Breast Cancer Dataset

The dataset used for this study consisted of breast cancer histology images obtained from the publicly available BreakHis dataset[87], which is widely recognised for research in breast cancer histopathology image classification. The BreakHis dataset contains microscopic images of breast tumour tissue, categorised into benign and malignant classes, further divided into specific subtypes. It comprises 7,909 histopathology images captured at different magnifications (40X, 100X, 200X, and 400X), ensuring a diverse representation of tissue samples.

The benign class includes subtypes such as: Adenosis, Fibroadenoma, Phyllodes tumour, and Tubular adenoma.

The malignant class is categorised into: Ductal carcinoma, Lobular carcinoma, Mucinous carcinoma, and Papillary carcinoma

The images in the dataset vary in resolution, ranging from 700 x 460 pixels to 1390 x 1104 pixels, providing detailed and high-resolution representations of breast cancer tissue. The diverse magnifications and tumour subtypes make the dataset highly suitable for training and evaluating CNNs for classification tasks.

3.2 Inspecting the Data

We began by inspecting the dataset, a crucial step in any image analysis problem. During this stage, we assessed the number of total images in the dataset and instances in each cancer type (malignant or benign). Hereafter, we calculated the number of images within each subtype of the malignant and benign groups. Extracting these values provided a clearer understanding of the dataset's composition, including class distribution and potential imbalances. This informed the preprocessing strategy by identifying the extent of necessary transformations such as normalization, augmentation, and down sampling to ensure a balanced and effective training dataset. In this case, we investigated the malignant class and its subtypes, followed by an in-depth investigation of the benign class. However, we found that the order of investigation was not important. We then inspected the quality

of the images – their resolution, and varying magnification to decipher how best to tackle variations in these variables (see Figure 5).

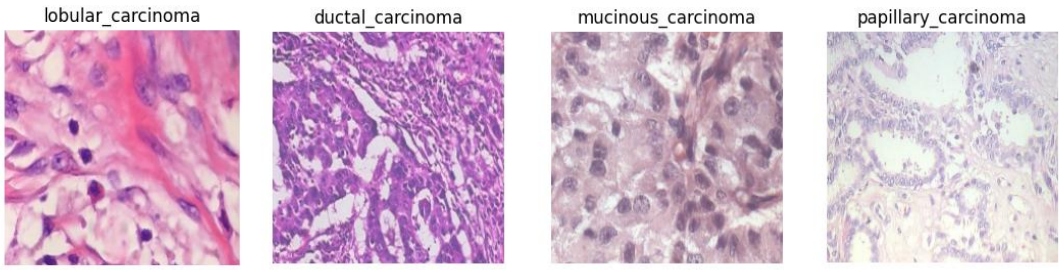


Figure 5: Sample of breast cancer histology images taken from the BreakHis dataset and associated subtype labels.

3.3 Processing the Data

3.3.1 Removing Irregularities

After an initial inspection, we moved on to the data processing stage. In this step, we removed all external noise to ensure the most important components of the dataset were isolated, since this would allow for successful application of the hand-crafted CNN. The first pre-processing step we applied was inspecting and removing irregular images. Using ImageLab[88] from the cleanvision library, all irregular images were removed. In this case, these were either odd-sized images (74 in the papillary carcinoma subtype, see Figure 7) or duplicates (4 from the lobular carcinoma subtype, see Figure 6). Once we removed these irregular images, we validated all images using ImageLab to ensure the dataset was clean and ready for further processing[89].

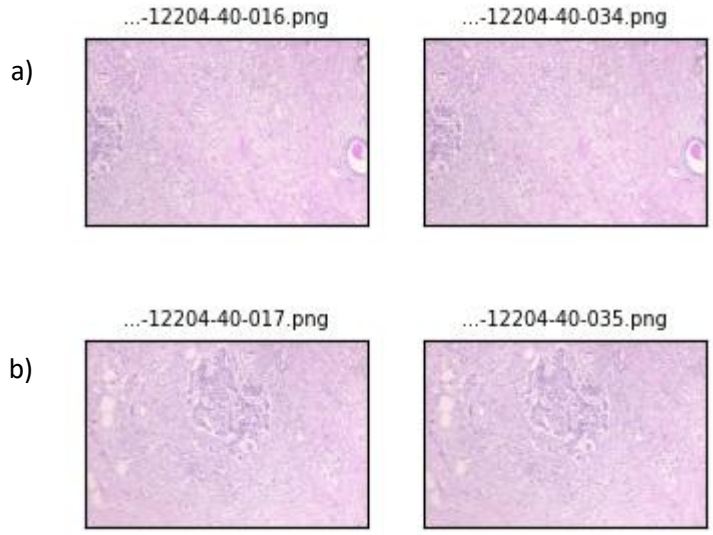


Figure 6: Examples of exact duplicate images from lobular carcinoma subtype.

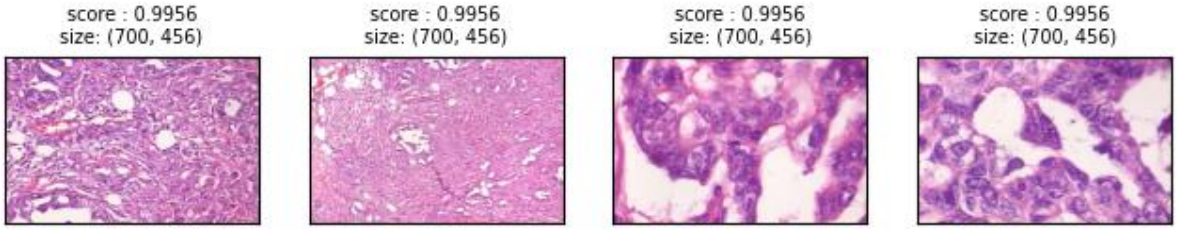


Figure 7: Example images from the papillary carcinoma subtype representing the most severe instances of odd image size (4 out of 74 total).

3.3.2 Defining the function in TensorFlow

We used TensorFlow to define the function for building a CNN [90, 91]. We then split the dataset into smaller subsets of histology images using stratified sampling techniques. In this process, images from each subtype were randomly sampled into training, testing and validation sets. This stratified sampling technique was applied with a split of 70% training (11568 images), 15% testing (336 images) and 15%

validation (336 images) to ensure a uniform distribution of breast cancer subtypes within each subset.

3.3.3 Data Normalisation

We standardised and normalised the images to account for variances in tumour heterogeneity, image features and staining techniques[92]. When defining the model parameters, we normalised the image size to 224 pixels. This ensured all images had the same square dimensions of 224 x 224 pixels x 3 channels [93]. This allowed the images to retain enough information for processing whilst reducing the number of training parameters and thus reducing the computational cost associated with this. We then normalised the images again, this time by scaling by 255 to convert the images, which are stored as 8-bit integers to floats between 0-1. This is the required model input for the CNN. Conversion to a float also significantly improves the required computation speed. The standardisation of images ensures the dataset is consistent yet contains an appropriate amount of variance[94].

We then separated all the images into batches (each containing 28 images) for fast and efficient processing. One of the benefits of batching the data in this way was that it allowed for short-term storage of the images, with individual image batches being removed from memory immediately after processing. Since the memory footprint of the entire image dataset was large, we needed to prioritise techniques that optimised CPU performance[95].

Due to the inherent stochasticity in experimental results of DL models, it is essential to set a random seed to ensure reproducibility and consistency in model training and evaluation. In the case of this project, we set the seed to 123. This ensured consistent reproducibility of the results when the different model configurations were evaluated over time[96]. This produced results that were deterministic and reproducible.

3.3.4 Data Augmentation

Since there was an imbalance in the number of images belonging to each subtype, it was necessary to down-sample those subtypes with the largest number of images (using a custom-written if-else conditional statement running in a loop) to equalise the numbers of images from each subtype which were then used for training[97]. We applied this to minimise the bias towards the larger class[98, 99]. As a result, the total number of images was reduced from 5367 (Figure 4)) to 2240, with 560 images belonging to each class (Figure 5)). This process is further illustrated in Figure 8 and Figure 9. We then applied data augmentation techniques to the images of each subtype/class to induce greater variability in the dataset. The specific data augmentation techniques we employed were random rotation, random brightness, and random zoom applications for the training images (Figure 10). As a result, data augmentation significantly reduced the overfitting of the initially larger classes.

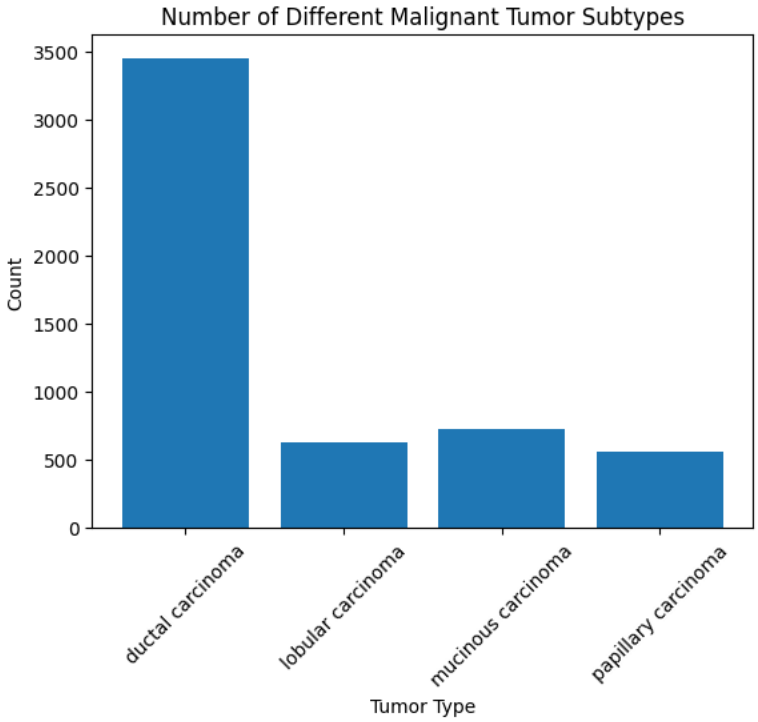


Figure 8: Distribution of malignant breast cancer images before down-sampling.

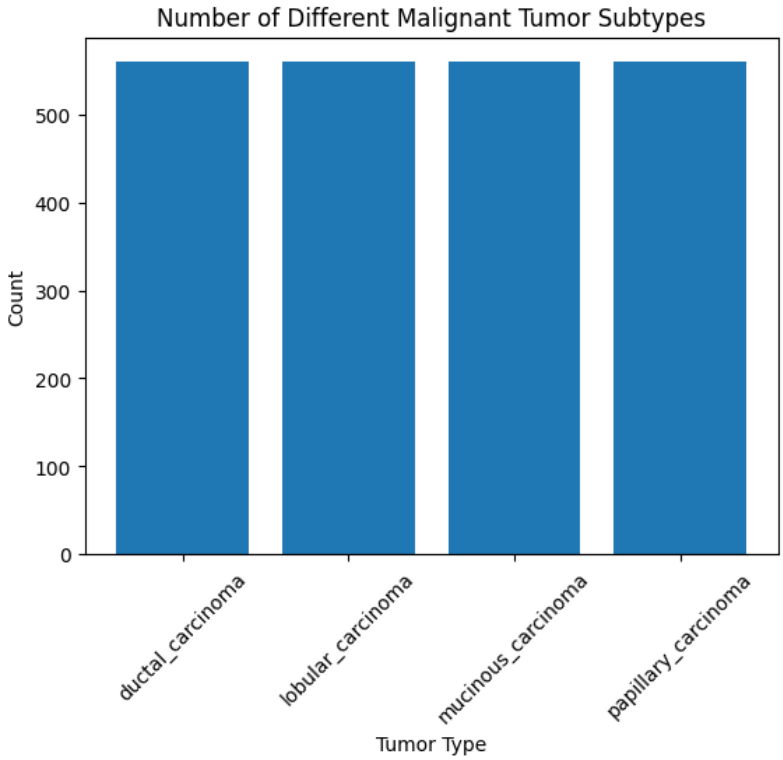


Figure 9: Distribution of malignant breast cancer images after down-sampling..

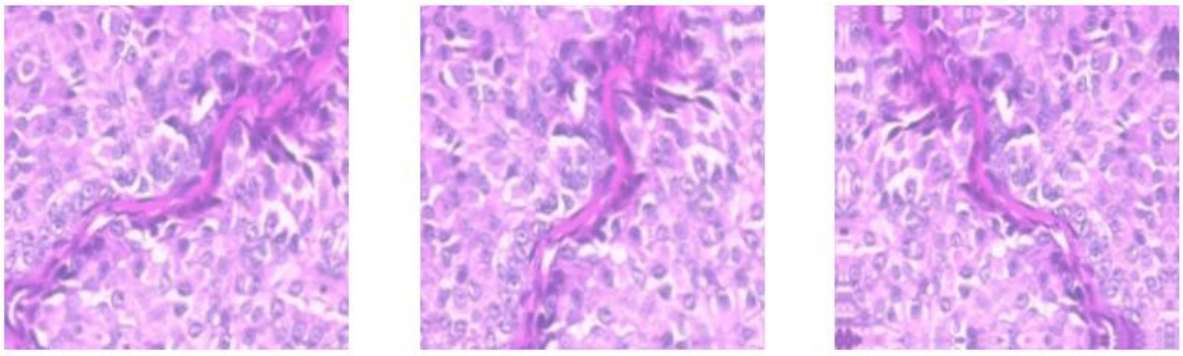


Figure 10: Example of data augmentation using random flip, random rotation and random zoom in the malignant breast cancer group.

3.3.5 Pre-processing Layers of the Model

CNNs have been shown to predict cancer subtypes with high levels of accuracy in the breast cancer field[100] and many other histological instances[101]. The pre-processing layers are specifically designed to prepare the input layers for the

convolutional layers, subsequently enabling easier extraction of key features within the model input. Pre-processing layers contribute to the overall standardisation of the input images, consequently enabling more efficient computation in the convolutional layers of the model, as well as improved overall model performance[102].

3.4 Defining the Custom CNN

The custom CNN architecture began with a pre-processing layer incorporating data augmentation and normalisation. The data augmentation layer introduced random rotations, brightness adjustments, and zooms, effectively increasing the variability of the training set and reducing the risk of overfitting. The normalisation layer then scaled pixel values to a range of 0 to 1, ensuring uniform input and faster convergence during training.

After pre-processing, the network consisted of three convolutional blocks. Each block included a convolutional layer, followed by an activation function (ReLU) and a max-pooling layer. The convolutional layers utilised filters (kernels) of size 3x3 to extract spatial features from the histology images. These filters identified patterns such as edges, textures, and shapes essential for distinguishing between breast cancer subtypes. The ReLU activation function introduced non-linearity, allowing the model to learn complex feature representations. Max-pooling layers (with a 2x2 pool size) followed each convolutional layer, reducing the spatial dimensions and the number of parameters, thus lowering the computational complexity and helping prevent overfitting.

After passing through the convolutional blocks, the output feature maps were flattened into a 1D vector, preparing the data for the fully connected layers. The flattened layer was followed by two fully connected (dense) layers. These layers further processed the extracted features to make predictions. A dropout layer was applied between the dense layers to randomly drop a portion of the neurons during training, which helped mitigate overfitting by improving generalisation.

The final layer of the network was a dense layer with a Softmax activation function responsible for outputting the probability distribution across the breast cancer

subtypes. This Softmax layer enabled the model to make multi-class classifications by assigning each input image to one of the predefined classes.

The network was optimised using the Adam optimiser, which effectively updated the model's weights during training, and the sparse categorical cross-entropy loss function was employed to handle multi-class classification tasks. This setup allowed the custom CNN to learn from the training data efficiently, achieving competitive performance in breast cancer subtype prediction.

3.5 Compile the Custom CNN Model

Loss functions (Sparse Categorical Cross Entropy and optimisation algorithms (Adaptive Moment Estimation (Adam)) were then applied to compile the model, whilst metrics including the accuracy, predictive values and F-score[103] were used to evaluate and monitor performance during the training period. The sparse categorical cross-entropy loss function is used for multiclass classification problems – it measures the logarithmic loss, which compares the predicted class probability to the actual observed probability, hereby penalising the probability based on how far off it is from the observed class probability, thus producing a loss measure[104,105]). Adam is a gradient-descent-based optimiser that minimises loss and improves performance during model training[106].

3.6 Model the Custom Training

We then initiated our first round of training by training the model using our hand-built custom CNN. Two more models, ResNet50 and EfficientNetB0, subsequently trained the dataset for transfer learning. We explain this later in the *ResNet50* and *EfficientNet* sections.

Using the training set, the Custom CNN was trained and validated. To ensure optimal model outcomes, we set the model to run over 20 epochs with early stopping and learning rate reduction on the plateau (ReduceLROnPlateau). Early stopping ensured that the required number of epochs were run during training whilst stopping training once there was no further improvement in validation accuracy and loss[107]. The same was true for the ReduceLROnPlateau function (from the Keras library); by optimising the model with this, we ensured that the learning rate was automatically

reduced when there was a plateau in model performance[108]. All epochs were completed to produce accuracy and loss scores for training and validation sets. Accuracy and Loss of both training and validation sets are plotted over all epochs.

3.7 Model Evaluation of the Custom CNN

The model was then evaluated by comparing the true observed scores (i.e. true subtype class) against the model predicted scores (predicted subtype class) and looking at the confusion matrices for each model. A confusion matrix is used to distinguish exactly where the model may be falling short in terms of classification[109]- it specifies exactly where the model is falling short in terms of classification and exactly what errors it is making. This includes false positives, for example, how many counts the model classifies an image as ductal carcinoma when it is, in fact, a papillary carcinoma.

3.8 Transfer Learning using ResNet50

The entire process was then repeated by applying transfer learning using pre-trained CNNs ResNet[110] and EfficientNet[111]. We applied it to the training set and validated it using the cross-validation set. We utilised ResNet because it has produced reliable accuracy, sensitivity, and specificity in multiclass image classification problems like the one at hand[112]. Furthermore, deep residual networks such as ResNet50 have a high computational efficiency while retaining good deep feature extraction[113, 114].

ResNet50 is comprised of a series of five stages, made up of a series of residual blocks which contain two 1 x 1 convolutional layers and one 3 x 3 convolutional layer each. What is unique about ResNet50 is that it makes use of skip connections between residual blocks to allow for easy gradient flow between layers during backpropagation and to enable identity mapping. In total ResNet50 consists of 49 convolutional layers with one final fully connected layer.

Following the convolutional layers, there is batch normalisation and activation functions (in this case ReLu).

3.9 Transfer Learning using EfficientNet

We also applied transfer learning using EfficientNet to the training set and validated using the cross-validation set. Similarly, EfficientNetB0 successfully classifies

images with high levels of accuracy whilst remaining computationally inexpensive. This, coupled with its ability to retain high levels of specificity, renders it appropriate for Transfer learning[115, 116].

EfficientNetB0 retains accuracy and computational efficiency by systematically scaling depth, width and resolution using a total of 82 layers. This compound scaling method allows the model to produce high performance whilst retaining low computational cost.

Importantly, EfficientNetB0 includes a series of mobile inverted bottleneck (MBConv) blocks with depth-wise convolutions[117]. Depth-wise convolutions are what enable the model to be computationally efficient- it applies separate filters to each input channel, followed by a 1 x 1 convolutional to combine the outputs thereby reducing the number of parameters. Above this, EfficientNet utilises swish activations which combine with the Sigmoid activation to improve gradient flow.

3.10 Evaluation of Trained Custom CNN, ResNet, and EfficientNet on the Test Set

Finally, the trained model was exposed to new unseen histology data and its performance was measured. The probabilities of successful final subtype classification were then calculated based on these measures.

After completing the training and evaluation process on the malignant breast cancer dataset, the same procedure was applied to the benign breast cancer dataset to ensure comprehensive model performance across different breast cancer subtypes. Both benign and malignant datasets were composed of histology images that required classification into their respective subtypes.

For the malignant dataset, the custom CNN was trained to distinguish between various malignant breast cancer subtypes, such as ductal carcinoma, lobular carcinoma, mucinous carcinoma, and papillary carcinoma. The model was tasked with learning the subtle differences between these aggressive forms of cancer through feature extraction and classification.

Once the model completed training on the malignant dataset, it was exposed to the benign breast cancer dataset, which contained subtypes like adenosis,

fibroadenoma, phyllodes tumour, and tubular adenoma. The same architecture and training protocol, including data augmentation, normalisation, convolutional blocks, and fully connected layers, were applied to this dataset. The goal was to evaluate the model's ability to generalise and accurately classify malignant and benign breast cancer subtypes.

By training the model on both datasets, it was possible to assess the CNN's robustness and adaptability in classifying histology images from different categories of breast cancer, allowing for a more holistic evaluation of its diagnostic potential. The results from both datasets were recorded, and comparisons were made to determine the model's strengths and weaknesses in distinguishing between benign and malignant subtypes, which differ in their biological behaviour and treatment implications.

3.11 Ethical Considerations

Our study involved secondary data analysis of histology images from the BreakHis dataset. The dataset is a collection of histology images sourced from the Prevention and Diagnosis (P&D) laboratory in Parana, Brazil between January 2014, and December 2014. These images were collected and analysed according to the ethical guidelines outlined by the institutional review board at the P&D Laboratory[118].

To protect the confidentiality and privacy of the research participants, all data identifiers were removed (anonymised) explicitly by the P&D laboratory. Participants were thus de-identified and only identified using abstract codes. This ensures that the identity of participants cannot be traced back to the original individuals. Additionally, our study strictly adhered to the principles of data anonymity and thus there was no attempt to re-identify participants in any way so that data confidentiality could be retained. The results of this study will be shared and published in accordance with the policies set out by the P&D laboratory.

Prior to the commencement of our research, this study received Ethics approval by the University of Cape Town, Faculty of Health Sciences, Human Research Ethics Committee (IRB00001938) (See Appendices). This ensured that our study complied

with all relevant ethical standards where applicable and that it continues to protect the rights and well-being of all participants involved.

The BreaKHis dataset is freely accessible to the public under the Creative Commons Attribution 4.0 International License (open source).

3.12 Prediction of Malignant Breast Tumours Using CNNs

3.12.1 Prediction of Malignant Breast Tumours Using the Custom CNN Model

The results for predicting malignant breast tumours using the custom CNN are outlined below. After training and validation, the model achieved a recorded validation accuracy of 64.58%, with a validation loss of 0.8785.

Upon evaluating the model's performance across epochs, the best results were observed at epoch 17. During this optimal epoch, the model recorded a validation accuracy of 64.58% and a validation loss of 0.8785 (Figure 11). The training accuracy for the same epoch was 62.97%, and the associated training loss was 0.8773. The custom CNN demonstrated an overall receiver operating characteristic area under the curve (ROC-AUC) score of 0.8660, indicating strong classification performance. This score suggests the model effectively distinguished the various malignant breast cancer subtypes despite the moderate accuracy levels.

These results demonstrate that the custom CNN performed reasonably well in predicting malignant tumour subtypes, and further tuning may improve both accuracy and classification robustness.

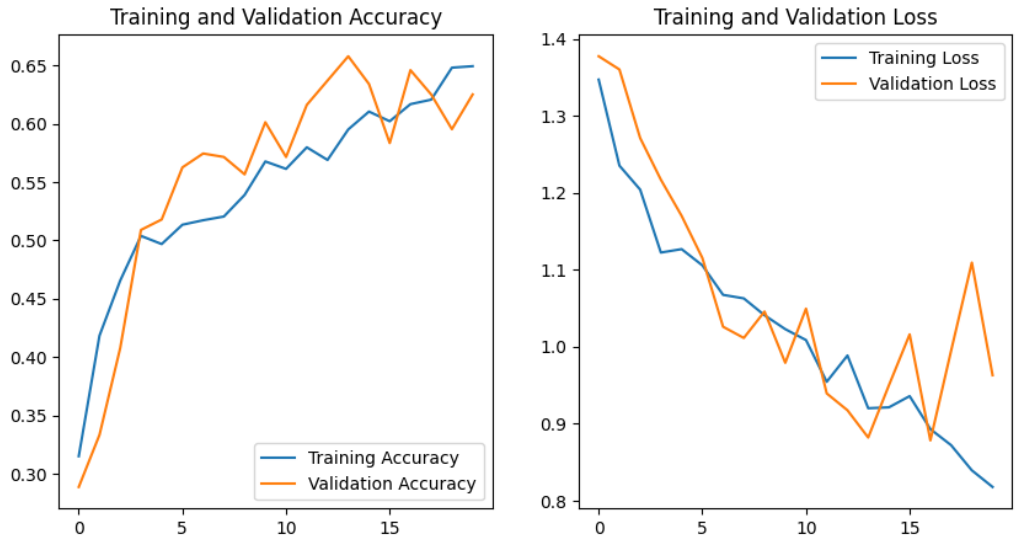


Figure 11: Accuracy and loss (training and validation) using custom CNN for malignant breast cancer images.

Whilst this observed best training and validation accuracy showed promise (greater than 60%), the significantly high loss showed that the model experienced some difficulty in extracting certain image features for correct classification. This error rate was better represented in the confusion matrix of Figure 12. Here, we observed a true positive classification count of 58 for ductal carcinoma, 52 for lobular carcinoma, 52 for mucinous carcinoma and 55 for papillary carcinoma. This also shows the subtypes for which a high misclassification count was observed. Overall, we observe that the model had a relatively good ability to accurately classify breast cancer histology images into malignant subtypes. The differences in results compared to existing literature can be attributed to several factors. First, the computational resources available for this study were limited, restricting the complexity and scale of hyperparameter tuning, data augmentation, and model training. Additionally, variations in dataset composition, preprocessing techniques, and model architectures across studies could contribute to performance differences. Finally, some existing studies may have leveraged ensemble methods or extensive transfer learning from larger, domain-specific datasets, which were beyond the scope of this MSc dissertation.

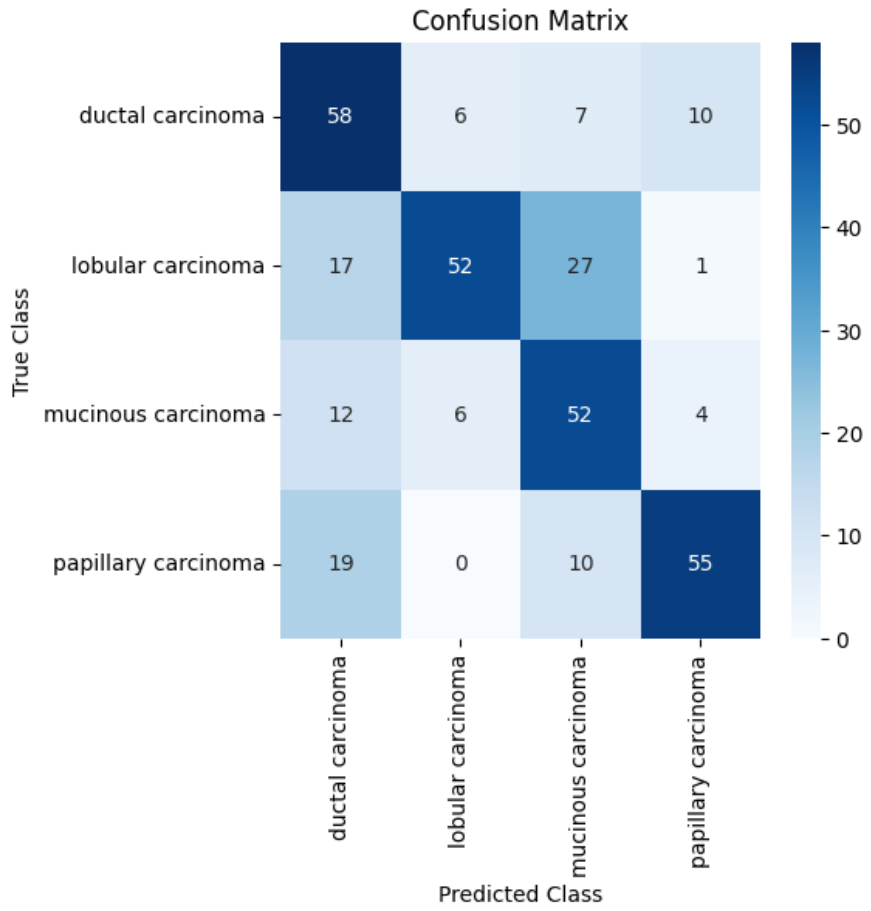


Figure 12: Confusion matrix for custom CNN for malignant breast cancer images.

The classification report in Table 1 outlines the precision, recall, F1-score and support scores for all subtypes. Here, we observed that lobular carcinoma is predicted with the greatest precision. Nevertheless, the ductal carcinoma subtype was also matched with a good recall of 0.72 and a high F1-score of 0.62. Notably, the papillary carcinoma subtype also recorded an excellent F1-score of 0.71. Adding to this, the weighted average for precision was 0.68, 0.65 for recall and 0.65 for F1-score, which suggests that the model performs well (see Table 2). These collected scores further prove that the custom CNN can accurately predict malignant breast cancer histology subtypes.

The discrepancies between our study's results and those of Majid et al. can be attributed to several key factors:

Data Augmentation Techniques: Majid et al. employed an intensive data augmentation pipeline, including advanced methods such as Generative Adversarial

Networks (GANs) to generate synthetic images, effectively addressing data imbalance and enriching the training set. In contrast, our study utilized standard augmentation methods, which may have limited the model's exposure to diverse data variations.

Classification Approach: While our study focused on multi-class classification of breast cancer subtypes, Majid et al. initially performed binary classification between benign and malignant tumors, achieving higher accuracy. They subsequently extended their approach to multi-class classification, leveraging the knowledge gained from the binary task to enhance performance. This staged approach likely contributed to their improved results.

Computational Resources: Implementing advanced augmentation techniques and training complex models require substantial computational resources. Our study faced limitations in this regard, restricting the scope of model complexity and the extent of data augmentation feasible within our computational constraints.

Table 1: Classification table for custom CNN for malignant breast cancer images

Subtype	Precision	Recall	F1-score	Support
Ductal carcinoma	0.55	0.72	0.62	81
Lobular carcinoma	0.81	0.54	0.65	97
Mucinous carcinoma	0.54	0.70	0.61	74
Papillary carcinoma	0.79	0.65	0.71	84

Table 2: Accuracy table for custom CNN for malignant breast cancer images

	Precision	Recall	F1-score	Support
Accuracy			0.65	336
Macro average	0.67	0.65	0.65	336
Weighted average	0.68	0.65	0.65	336

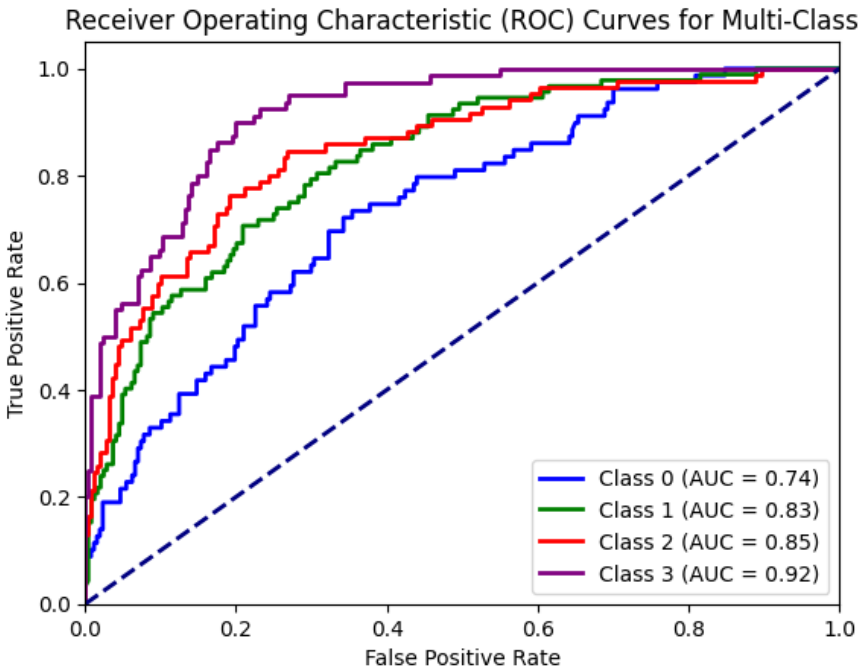


Figure 13: ROC curves for custom CNN for malignant breast cancer subtypes. Class 0 represents ductal carcinoma, class 1 is lobular carcinoma, class 2 mucinous carcinoma, and class 3 is papillary carcinoma.

The ROC curves and area under the curve (AUC) scores for each class is reported in Figure 13 as follows: The class which observed the greatest AUC and subsequent best classification by our custom CNN was papillary carcinoma with an AUC of 0.92. Both mucinous carcinoma and lobular carcinoma followed closely with AUC scores of 0.85 and 0.83, respectively. Unsurprisingly, as observed in the classification matrix, ductal carcinoma received the lowest AUC of 0.74 and was thus, the subtype with the highest misclassification. The differences between our study and that of Majid et al. can be attributed to variations in data augmentation strategies, classification approach, and computational constraints. Majid et al. employed advanced augmentation techniques, including Generative Adversarial Networks (GANs), to generate synthetic data and address class imbalances, whereas our study relied on standard augmentation methods. Additionally, their study initially focused on binary classification before extending to multi-class classification, allowing for an incremental learning approach that may have improved accuracy. Furthermore, differences in computational resources likely played a role, as more extensive data augmentation and longer training times require significant hardware capabilities, which were limited in our study. These factors collectively contribute to the observed discrepancies in model performance.

3.12.2 Prediction of Malignant Breast Tumours Using the ResNet Model

Following the evaluation of our custom CNN, the images were subsequently exposed to the well-known ResNet50 model as part of the transfer learning approach. This allowed for a direct comparison between the custom CNN model and the widely recognised ResNet50 model. Early stopping was engaged at epoch 15 to prevent overfitting.

At the optimal epoch (epoch 9), the ResNet50 model achieved an impressive training accuracy of 97.00%, with a minimal training loss of 0.1122. In the validation phase, the model recorded an accuracy of 77.48% and a validation loss of 0.5482.

These results indicate that the ResNet50 model demonstrated superior classification performance, consistently achieving high accuracy in training and validation while maintaining low loss levels. Its stable performance metrics further reinforced the model's ability to accurately classify malignant breast cancer subtypes (Figure 14). This suggests that ResNet50 is a highly effective model for distinguishing between different histological subtypes of malignant breast cancer, offering a reliable and efficient alternative for clinical applications.

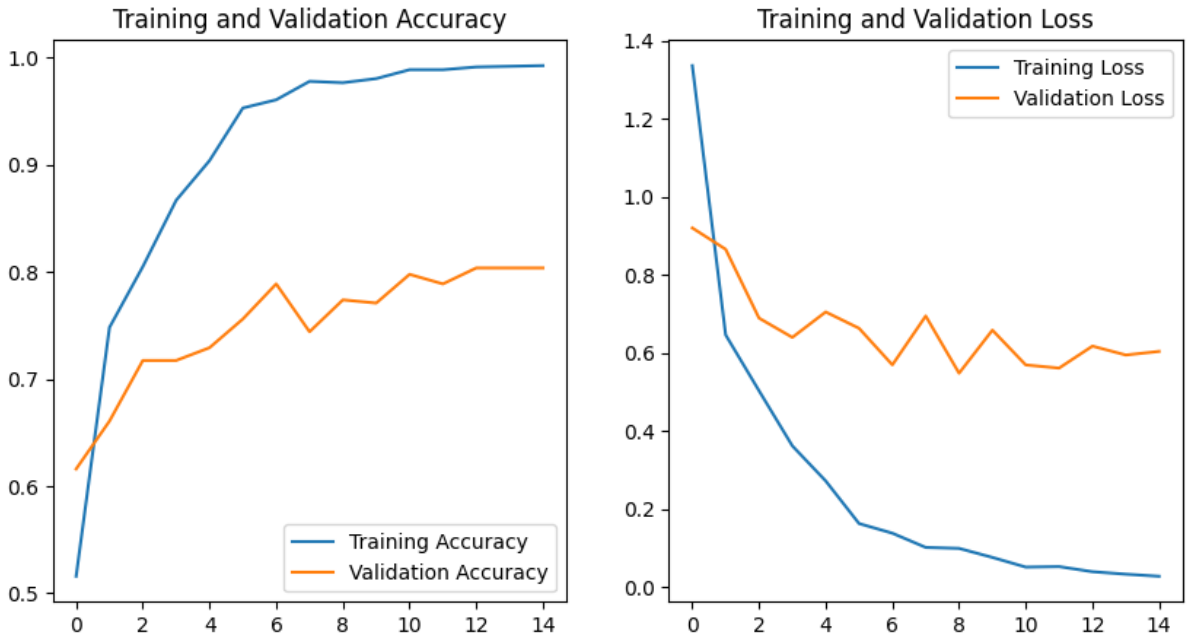


Figure 14: Accuracy and loss (training and validation) using ResNet50 for malignant breast cancer images.

ResNet' s ability to accurately classify BreakHis breast cancer images into subtypes is further proven upon inspection of Figure 15. The number of images that were misclassified was significantly reduced and as a result, we observed few false positives and false negatives. Notably in this model, the produced ROC-AUC score was 0.9234. This coupled with the high validation accuracy score emphasises that ResNet is good at subtype classification and can effectively differentiate between classes in malignant breast cancer.

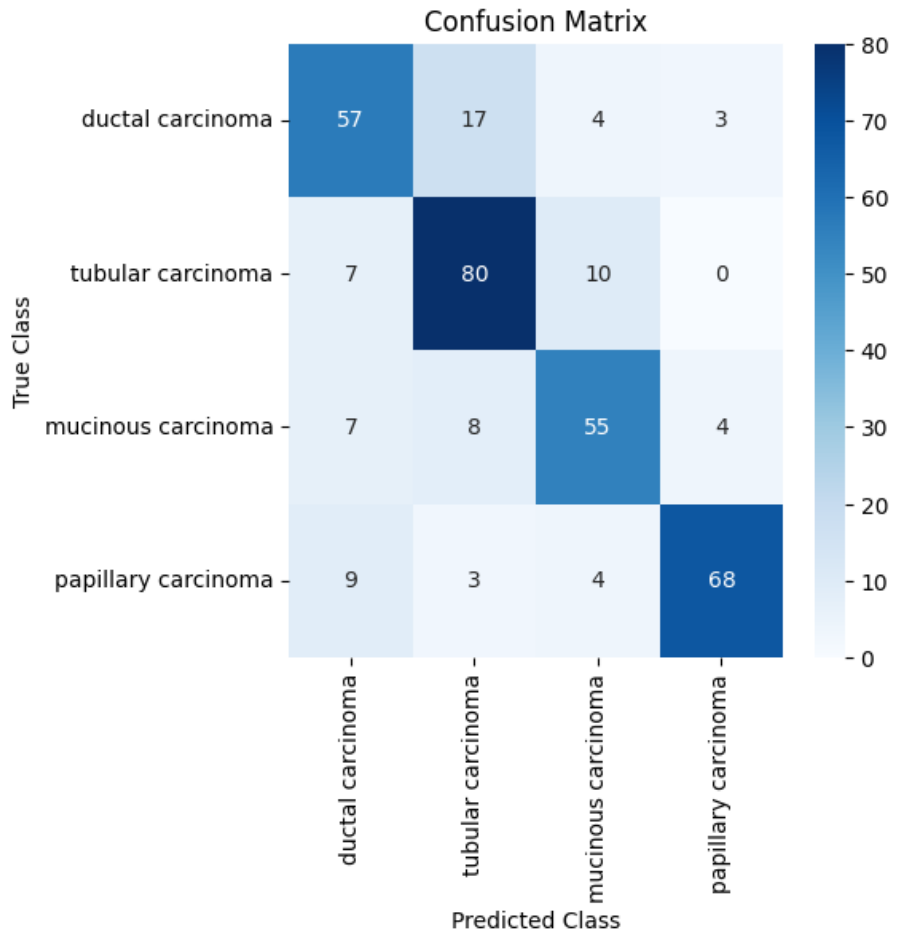


Figure 15: Confusion matrix for ResNet model for malignant breast cancer images.

Table 3 showed that the overall precision score is very good for each subtype, with very little variation between subtypes (ductal carcinoma observed the lowest precision at 0.71). Similarly for recall and F1-score- the measures for each subtype were high and well supported. Subsequently, Table 4 shows the overall weighted average for precision, recall and F1-score as 0.78, 0.77, 0.77. This is indicative of the model's strong classification ability – that it performed at a high level and was excellent at discerning different malignant breast cancer subtypes from our image dataset.

Table 3: Classification table for ResNet50 for malignant breast cancer images

Subtype	Precision	Recall	F1-score	Support
Ductal carcinoma	0.71	0.70	0.71	81
Tubular carcinoma	0.74	0.82	0.78	97
Mucinous carcinoma	0.75	0.74	0.75	74
Papillary carcinoma	0.91	0.81	0.86	84

Table 4: Accuracy table for ResNet50 for malignant breast cancer images

	Precision	Recall	F1-score	Support
Accuracy			0.77	336
Macro average	0.78	0.77	0.77	336
Weighted average	0.78	0.77	0.77	336

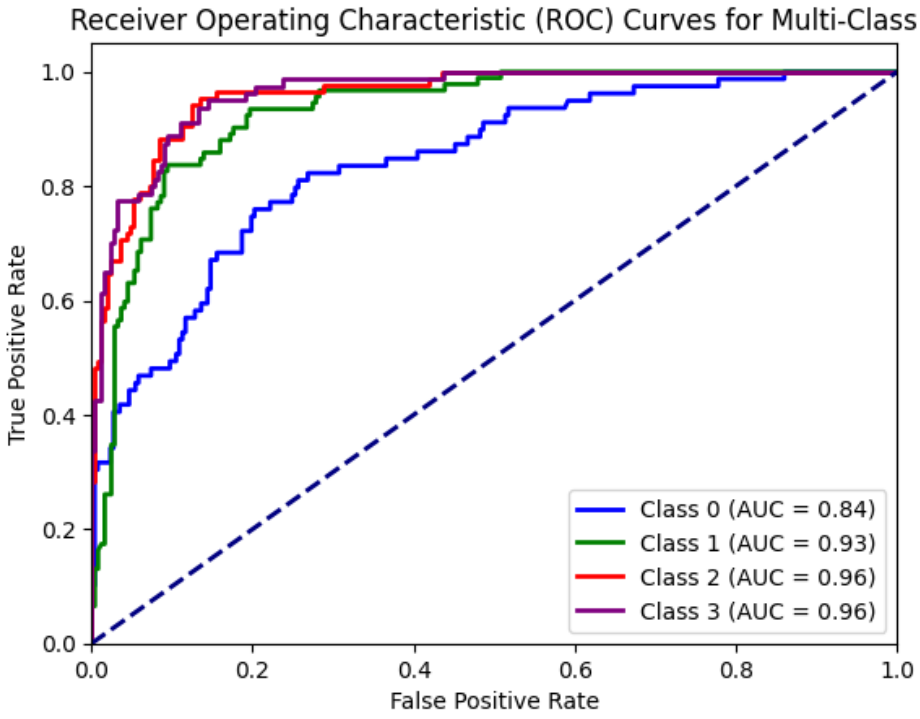


Figure 16: ROC curves for ResNet50 for malignant breast cancer images

The ROC curves and area under the curve (AUC) scores for each class further support that the model is superior at classifying breast cancer images into subtypes. Papillary carcinoma and mucinous carcinoma observed the highest AUC of 0.96. This high AUC is continued across subtypes with a reported AUC of 0.93 for lobular carcinoma followed by 0.84 for ductal carcinoma, which was once more the subtype with the highest misclassification.

3.12.3 Prediction of Malignant Breast Tumours Using the EfficientNetB0 Model

The results for predicting malignant breast tumours using the EfficientNetB0 are detailed below. After training and validation, the model achieved a recorded validation accuracy of 62.5%, with a validation loss of 0.8981. EfficientNetB0's performance was evaluated across epochs. In this case, early stopping was observed at epoch 6 and the best epoch performance was at epoch 1. At this epoch, a training accuracy of 43.30% and training loss of 1.3232 was recorded. An overall

ROC score of 0.8327 was demonstrated by the model, hereby suggesting that despite the relatively moderate accuracy, the model was still able to effectively distinguish between the various malignant breast cancer subtypes. Whilst this ROC is the lowest of our three models, it is still supports that EfficientNetB0 is a competitive image classification model.

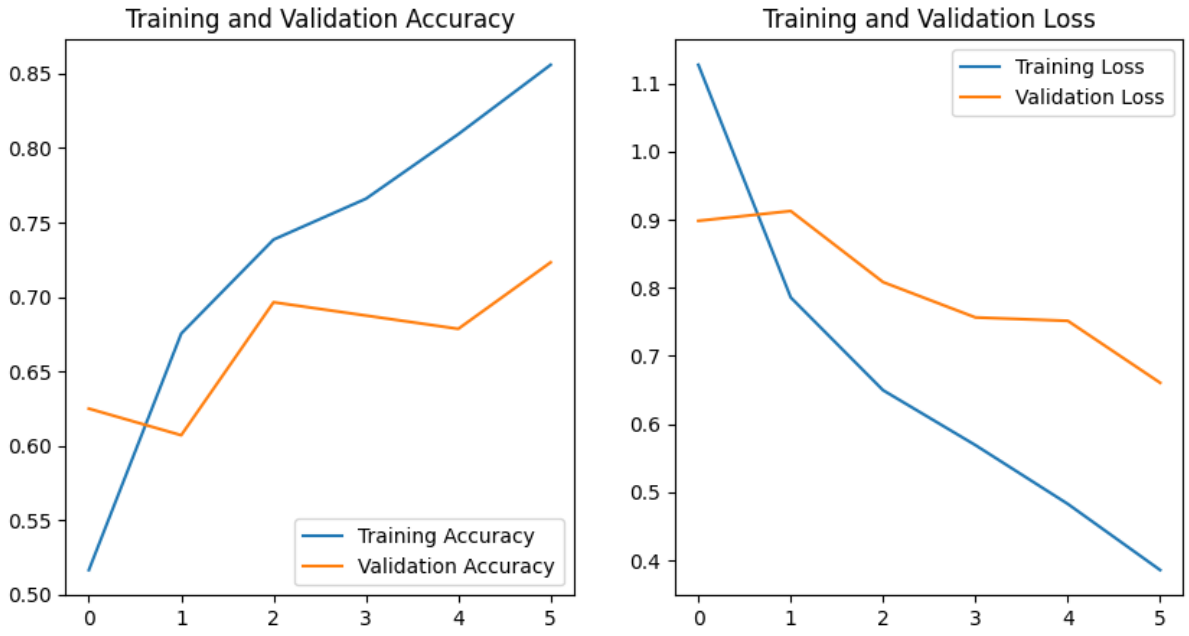


Figure 17: Accuracy and loss (training and validation) using EfficientNetB0 for malignant breast cancer images.

Figure 18 shows the confusion matrix for EfficientNetB0. Whilst its ability to discern between subtypes was not as good as the other two models, it still produced a strong classification score for each subtype. Notably however, classification bias towards ductal carcinoma subtype was once again observed and the highest of all our models.

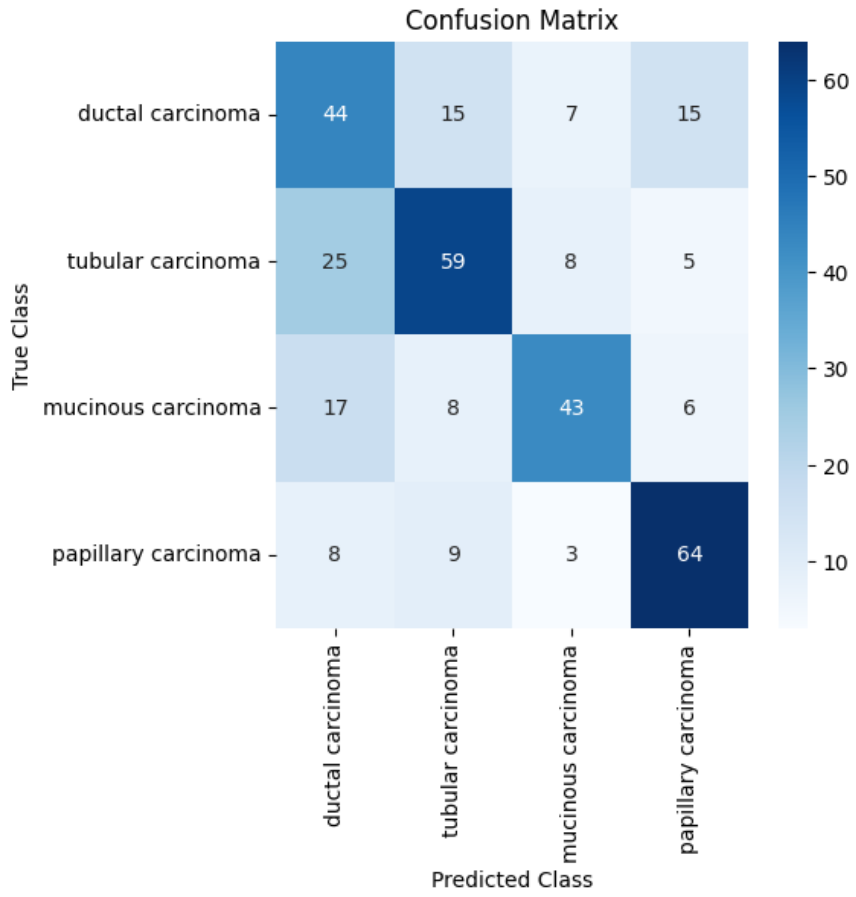


Figure 18: Confusion matrix for EfficientNetB0 for malignant breast cancer images.

The Classification Report for EfficientNetB0 showed that the overall precision, recall and F1-score was moderate. This supports the notion that it produced weaker performance in comparison to our custom CNN and ResNet50. Nevertheless, we observed that in this case, the papillary carcinoma subtype showed the highest precision with a score of 0.71. Papillary carcinoma also produced the highest F1-score of all the subtypes at 0.74. Table 6 highlights the weighted averages of each measure; 0.63, 0.62 and 0.63 for precision, recall and F1-score. This emphasises once more the model's weaker performance in successfully classifying breast cancer histology images into subtypes.

Table 5: Classification report for EfficientNetB0 for malignant breast cancer images

Subtype	Precision	Recall	F1-score	Support
Ductal carcinoma	0.47	0.54	0.50	81
Tubular carcinoma	0.65	0.61	0.63	97
Mucinous carcinoma	0.70	0.58	0.64	74
Papillary carcinoma	0.71	0.76	0.74	84

Table 6: Accuracy table for EfficientNetB0 for malignant breast cancer images

	Precision	Recall	F1-score	Support
Accuracy			0.62	336
Macro average	0.63	0.62	0.63	336
Weighted average	0.63	0.62	0.63	336

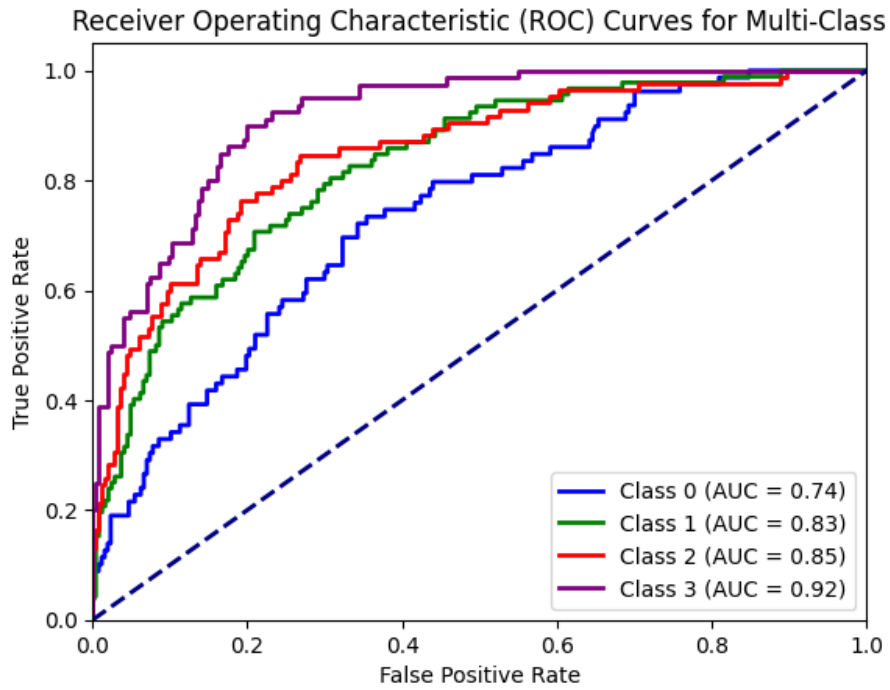


Figure 19: ROC curves for EfficientNetB0 for malignant breast cancer images

The AUC scores for each subtype were recorded in the ROC curves in Figure 19. Notably, these scores were the same as determined for our custom CNN; in other words, the custom CNN correctly distinguished images between subtypes at the rate as that of EfficientNetB0 with an AUC of 0.92 for papillary carcinoma, 0.85 for mucinous carcinoma, 0.83 for lobular carcinoma and 0.74 for ductal carcinoma.

3.13 Prediction of Benign Breast Tumours Using CNNs

3.13.1 Prediction of Benign Breast Tumours Using the Custom CNN Model

After exposing our benign breast cancer images to our custom CNN, we observed the following: the best epoch (epoch 20) produced a validation accuracy of 66.92% with an associated validation loss of 0.8350. Notably, the observed ROC-AUC was 0.8984. This good ROC-AUC value combined with the good validation accuracy of 66.92% shows that the custom CNN exhibited classification performance.

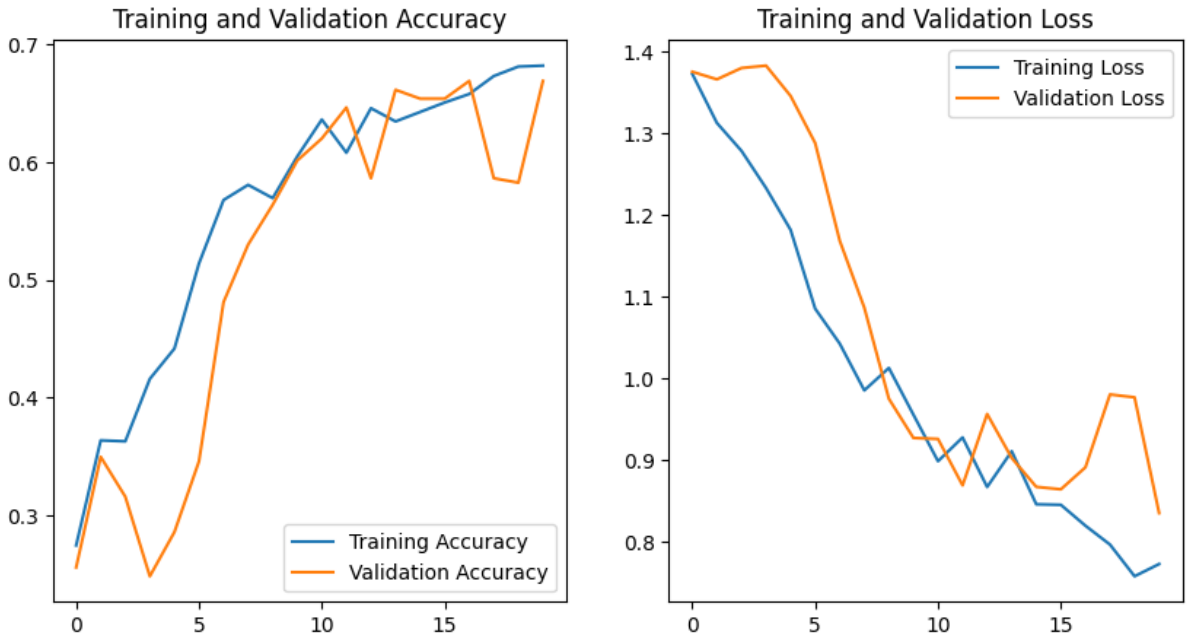


Figure 20: Accuracy and loss (training and validation) using custom CNN for benign breast cancer images.

Figure 21 supports that the model performs well at benign breast cancer subtype classification, however there was a small bias towards misclassification of fibroadenoma by our custom CNN. The custom CNN produced true positive counts of 51 for adenosis, 17 for fibroadenoma, 49 for phyllodes tumour and 61 for tubular adenoma.

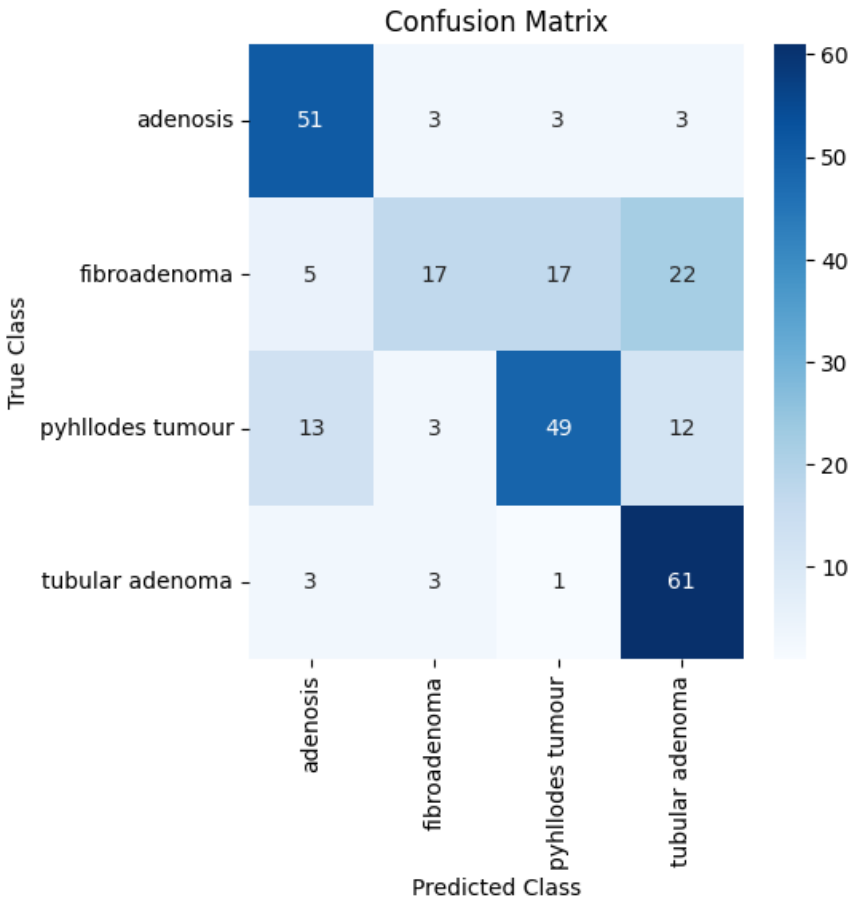


Figure 21: Confusion matrix for custom CNN for benign breast cancer images.

Table 7 shows the classification report for our custom CNN on benign breast cancer images. Evidently, this supports that our custom CNN boasts moderate model performance. Adenosis and tubular adenoma subtypes show the greatest success. Notably, the model showed the worst performance when classifying fibroadenoma. This showed that despite the overall moderate model performance, bias was exhibited towards the fibroadenoma subtype.

Table 7: Classification report for custom CNN for benign breast cancer images

Subtype	Precision	Recall	F1-score	Support
Adenosis	0.71	0.85	0.77	60
Fibroadenoma	0.65	0.28	0.39	61
Phyllodes Tumour	0.70	0.64	0.67	77
Tubular Adenoma	0.62	0.90	0.73	68

Table 8: Accuracy table for custom CNN for benign breast cancer images

	Precision	Recall	F1-score	Support
Accuracy			0.67	266
Macro average	0.66	0.66	0.64	266
Weighted average	0.67	0.67	0.64	266

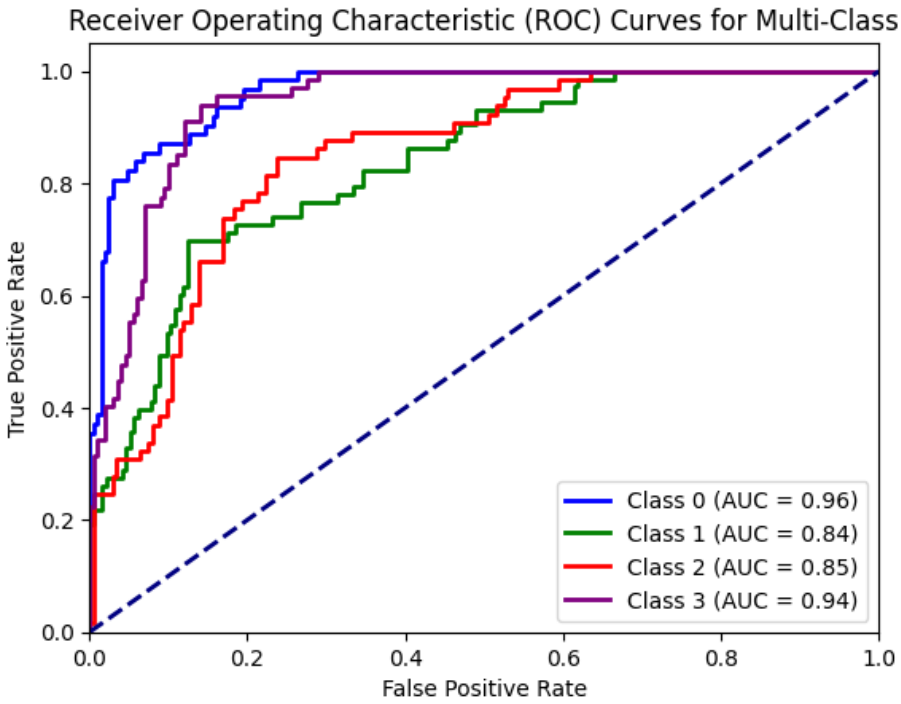


Figure 22: ROC curves for custom CNN for benign breast cancer images

The overall accuracy (67%) and high ROC-AUC scores (0.94 for tubular adenoma, 0.85 for phyllodes tumour, 0.84 for fibroadenoma and 0.96 for adenosis) combined suggest that the custom CNN can accurately classify benign breast cancer images into subtypes at a high level.

3.13.2 Prediction of Benign Breast Tumours Using the ResNet Model

When we applied ResNet to our benign breast cancer images, early stopping was initiated at epoch 10. The best model performance was recorded at epoch 4 which produced a validation accuracy of 77.06% and validation loss of 0.5247. This suggests that ResNet shows superior subtype classification performance for benign breast cancer images (see Figure 24). This accompanied with the strong reported ROC-AUC score of 0.9552 renders ResNet exceptional at benign breast cancer classification.

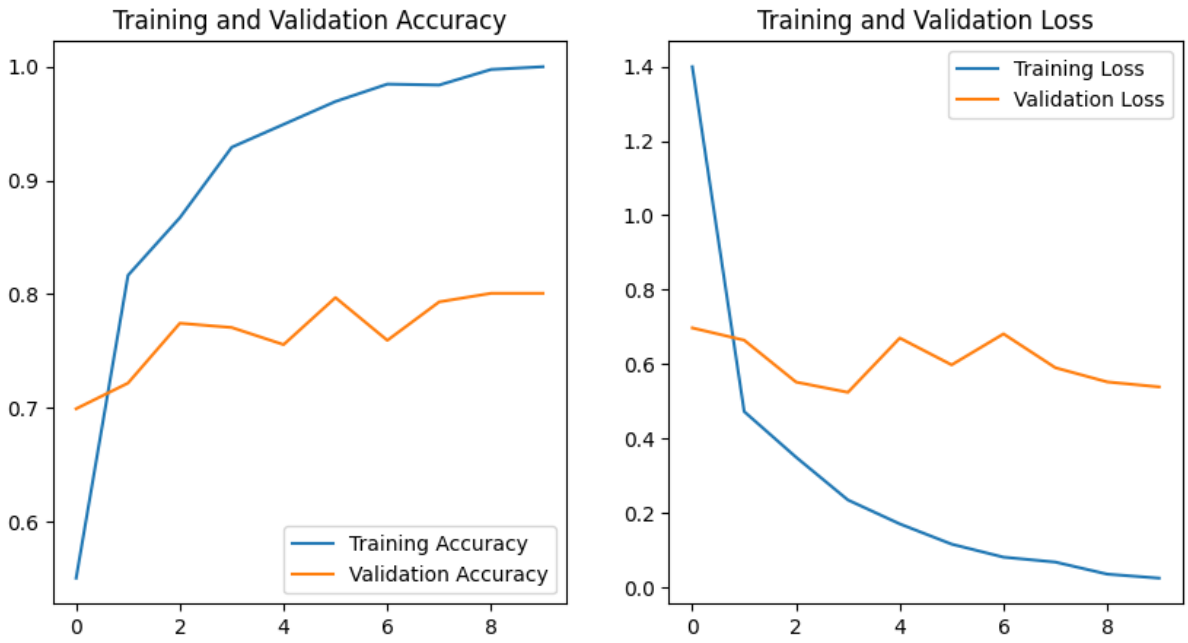


Figure 23: Accuracy and loss (training and validation) using ResNet50 for benign breast cancer images.

Figure 25 shows the true classification count for each subtype. Here we observed very few misclassifications to accompany the high accuracy and ROC-AUC scores for the ResNet50 model. Tubular adenoma produced an AUC of 0.99, phyllodes tumour 0.94, fibroadenoma 0.90 and adenosis with an AUC score of 0.99 (see Figure 26). Overall, the performance was significantly better than that of our custom CNN. ResNet was more robust during classification and produced fewer classification errors overall-this may be a result of overfitting of ResNet.

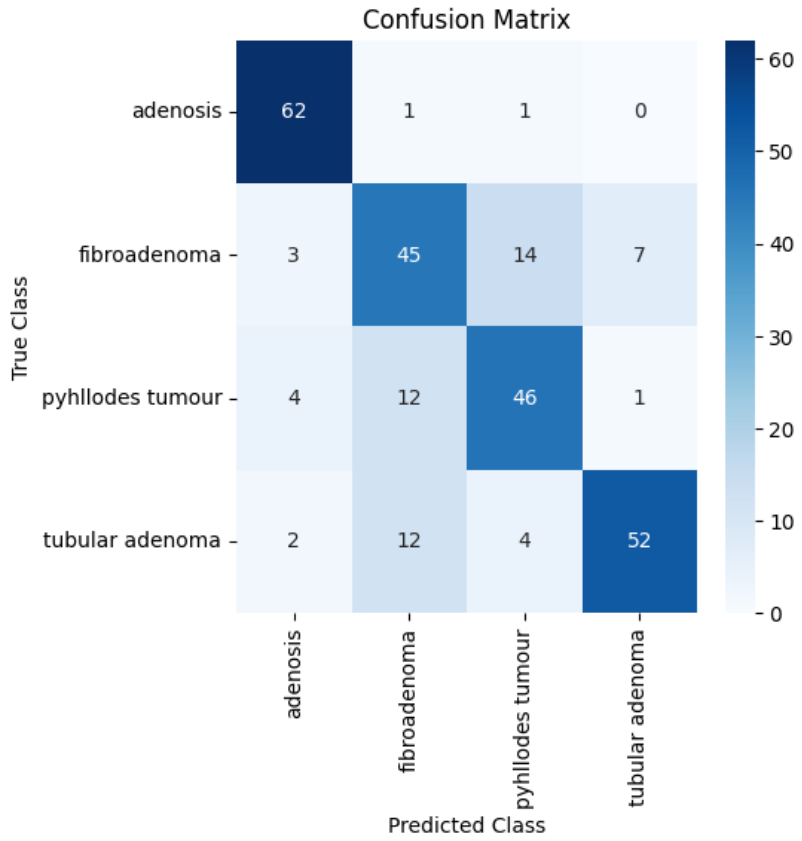


Figure 24: Confusion matrix for ResNet50 for benign breast cancer images.

The classification table for ResNet50 suggests that ResNet50 is an excellent model for benign breast cancer subtype classification. Precision, recall and F1-scores for each subtype are all high, however lowest in fibroadenoma with 0.64 precision, 0.65 recall and 0.65 F1-score. Furthermore, the weighted averages of 0.77 for precision, 0.77 recall and 0.77 F1-score is further evidence to support that ResNet can accurately classify benign breast cancer subtypes from histology images.

Table 9: Classification report for ResNet50 for benign breast cancer images

Subtype	Precision	Recall	F1-score	Support
Adenosis	0.87	0.97	0.92	64
Fibroadenoma	0.64	0.65	0.65	69
Phyllodes Tumour	0.71	0.73	0.72	63
Tubular Adenoma	0.87	0.74	0.80	70

Table 10: Accuracy table for ResNet50 for benign breast cancer images

	Precision	Recall	F1-score	Support
Accuracy			0.77	266
Macro average	0.77	0.77	0.77	266
Weighted average	0.77	0.77	0.77	266

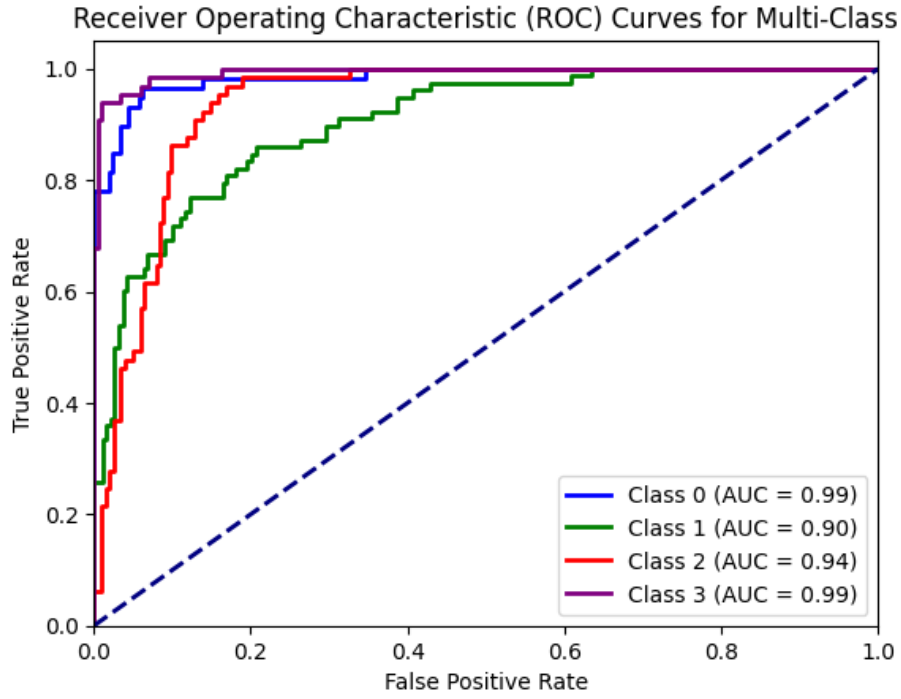


Figure 25: ROC curves for ResNet50 for benign breast cancer images

3.13.3 Prediction of Benign Breast Tumours Using the EfficientNet Model

We then proceeded to expose our benign breast cancer images to EfficientNet. Early stopping was observed at epoch 6, and the best epoch was recorded at epoch 1. At epoch 1 we observed a validation accuracy of 65.79% and a validation loss of 0.7862. For this same epoch, a training accuracy of 49.91% and validation loss of 1.1754 was observed. This moderate validation accuracy alongside a ROC-AUC score of 0.89422 suggests that EfficientNetB0 can effectively classify benign breast cancer images into distinct subtypes with moderate accuracy.

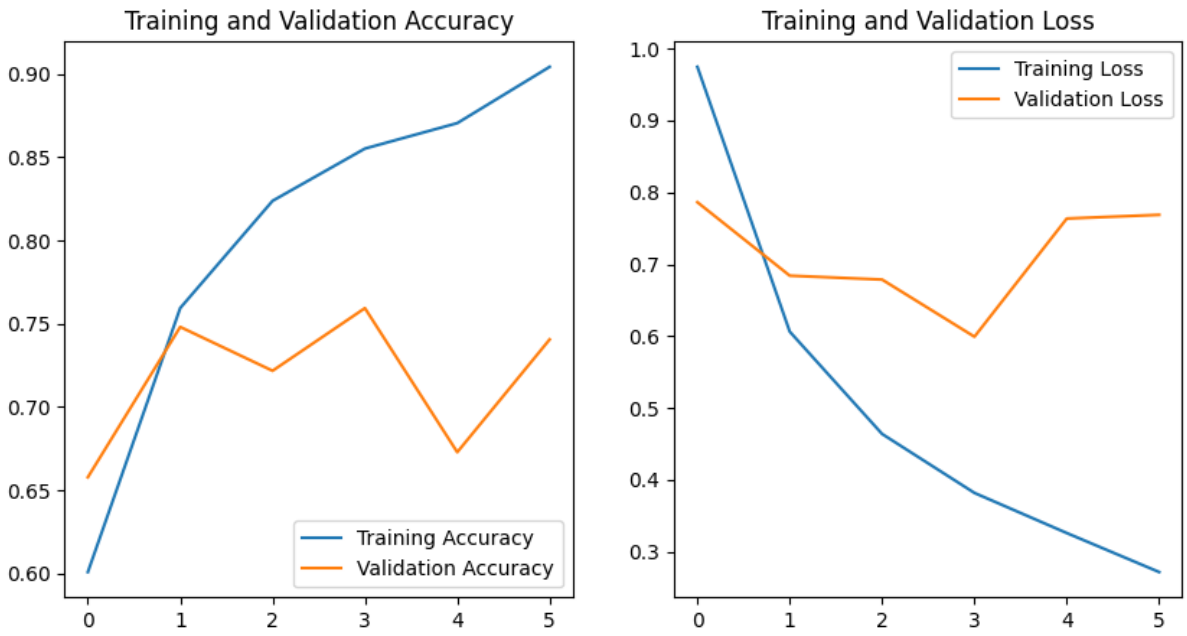


Figure 26: Accuracy and loss (training and validation) using ResNet50 for benign breast cancer images.

Figure 27 shows the misclassification rate for EfficientNetB0. Evidently, the model performed quite well, with a couple of errors. Nevertheless, it is noticeable that majority of misclassification occurred with the fibroadenoma subtype, suggesting that there is bias against this class. Overall, however, the classification matrix shows good classification for most subtypes with a true positive count of 55 for adenosis, 41 for fibroadenoma, 35 phyllodes tumour and 44 tubular adenoma.

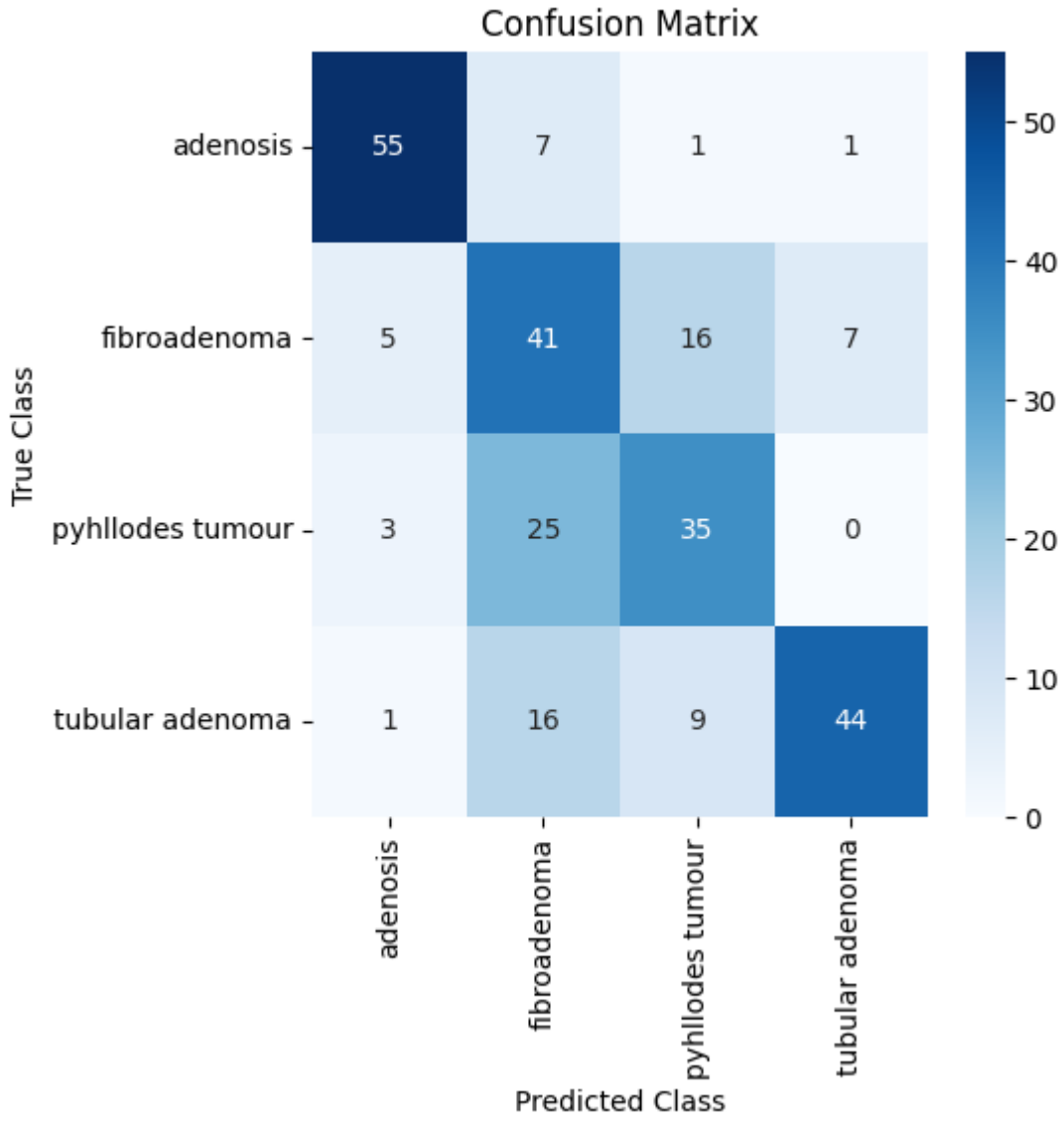


Figure 27: Confusion matrix for EfficientNetB0 for benign breast cancer images.

Table 11 confirms that EfficientNetB0 produced good performance. According to the classification report, fibroadenoma experiences the poorest classification – this is reflected in the low precision score of 0.46 and its low F1-score of 0.52. The weighted averages for precision, recall and F1-score for EfficientNetB0 are also sufficient with 0.68 for precision, 0.66 recall, 0.66 F1-score. This is further evidence that EfficientNetB0 is capable of accurately predicting benign breast cancer subtypes from histology images.

Table 11: Classification report for ResNet50 for benign breast cancer images

Subtype	Precision	Recall	F1-score	Support
Adenosis	0.86	0.86	0.86	64
Fibroadenoma	0.46	0.59	0.52	69
Phyllodes Tumour	0.57	0.56	0.56	63
Tubular Adenoma	0.85	0.63	0.72	70

Table 12: Accuracy table for ResNet50 for benign breast cancer images

	Precision	Recall	F1-score	Support
Accuracy			0.66	266
Macro average	0.68	0.66	0.67	266
Weighted average	0.68	0.66	0.66	266

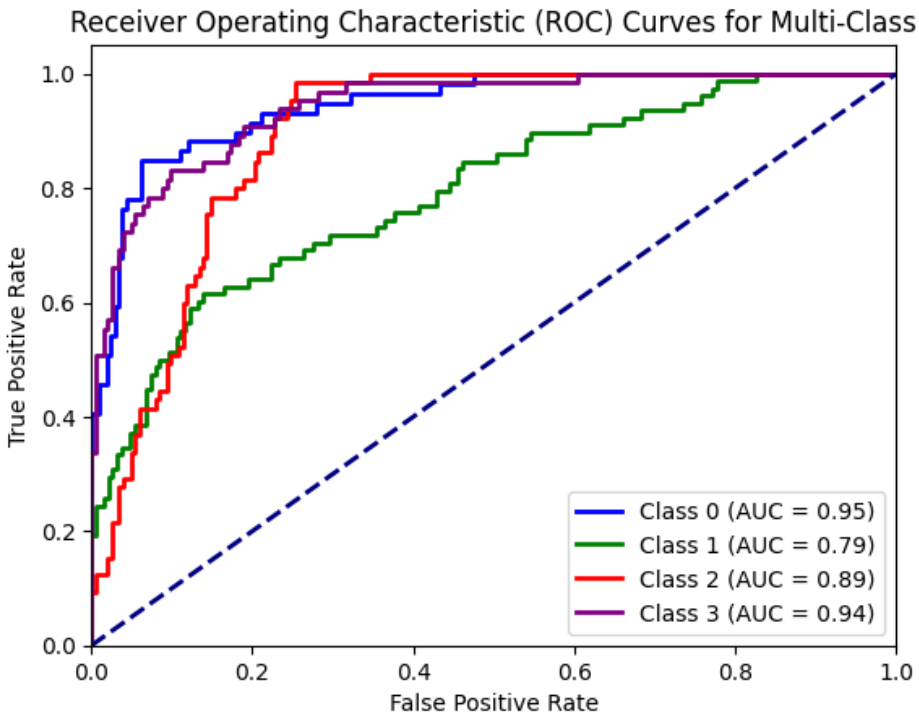


Figure 28:

The ROC Curves for each subtype are further evidence that EfficientNet can aptly classify benign breast cancer images into subtypes. As observed in Figure 27, we see that the fibroadenoma subtype produced the lowest AUC score at 0.79. Tubular adenoma produced an AUC of 0.94, AUC for phyllodes tumour was reported as 0.89 and adenosis produced the highest AUC at 0.95. This combined with the strong ROC-AUC and moderate accuracy renders EfficientNetB0 effective at subtype classification of benign breast cancer images.

3.14 Comparative Results

3.14.1 Malignant Breast Cancer

After individual examination of each model's performance, it was imperative to compare across models. Table 13 illustrates this well; here we observe the differences in selected performance metrics for malignant breast cancer classification. It was noted that ResNet outperformed both models in every metric and is overall the best model for malignant breast cancer subtype prediction. Importantly, it produced a strong validation accuracy of 77% with a small loss of 0.55

(Remember we aim to maximise model accuracy and minimise loss). Similarly, our custom CNN outperformed EfficientNetB0 in every metric, and produced a competitive accuracy of 65% and a good ROC-AUC of 0.86.

Table 13: Model performance for malignant classification across custom CNN, ResNet50 and EfficientNetB0.

	Custom CNN	ResNet50	EfficientNetB0
Validation Accuracy	0.65	0.77	0.63
Validation loss	0.88	0.55	0.90
Precision (weighted average)	0.68	0.78	0.63
Recall (weighted average)	0.65	0.77	0.62
F1-Score (weighted average)	0.65	0.77	0.63
ROC-AUC	0.86	0.92	0.83

3.14.2 Benign Breast Cancer

In the case of benign breast cancer images, it was observed that as in the malignant case, ResNet50 was the best performing model with the highest ROC-AUC and validation accuracy of all models at 0.96 and 77% respectively. It also produced the lowest validation loss of all models, with a score of 0.52. Notably, EfficientNetB0 showed improved performance for benign breast cancer subtypes as compared to its performance when exposed to malignant breast cancer images – all metrics were bettered, and it produced an excellent ROC-AUC of 0.89. Our custom CNN also showed good performance- its ability to discern benign subtypes from histology images was commendable. In fact, its performance was better than that when exposed to malignant breast cancer images with an improved ROC-AUC of 0.90 as compared to 0.86 for malignant.

Table 14: Model performance for benign classification across custom CNN, ResNet50 and EfficientNetB0.

	Custom CNN	ResNet50	EfficientNetB0
Validation Accuracy	0.67	0.77	0.66
Validation loss	0.84	0.52	0.79
Precision (weighted average)	0.67	0.77	0.68
Recall (weighted average)	0.67	0.77	0.66
F1-Score (weighted average)	0.67	0.77	0.66
ROC-AUC	0.90	0.96	0.89

Overall, all the models that our both our image datasets (benign and malignant) were exposed to produced good results for subtype classification, solidifying the importance of transfer learning in image recognition tasks. Furthermore, the differences in performance between malignant and benign groups shows that as a whole, malignant cancers are more difficult to distinguish between, and research efforts should be especially focused on these.

4 Discussion

This thesis has aptly discussed the role of machine learning in breast cancer care, whilst also delving into the significance behind integrating AI into healthcare. Harnessing the power of machine learning is the forefront of healthcare innovation[119] [120]. The integration of machine learning algorithms into cancer care not only represents the future of oncology research but also holds profound implications for cancer patient care[121].

Perhaps the biggest challenge in breast cancer diagnosis and treatment is the timely identification of the disease. Using machine learning techniques in this regard enables clinicians to analyse large amounts of data, such as histology images and mammograms from cancer data repositories such as The Cancer Genome Atlas (TCGA) database[122], with significant efficiency and accuracy. The ability to process and interpret complex datasets at scale in this way thus has the potential to significantly reduce breast cancer diagnosis times, leading to earlier intervention and improved patient outcomes[123, 124].

Additionally, by improving the speed and accuracy of cancer diagnosis, we can streamline the treatment pipeline. Planning treatment strategies and starting treatment may become significantly easier and quicker to achieve. This would subsequently enable patients to access tailored therapeutic options more quickly. This not only improves the efficacy of cancer treatments but also allows for a more patient-centred approach to care, where individual needs and preferences are prioritised. Examples of this includes projects such as the IMB Watson for Oncology (WFO)[125], Google Health DeepMind[126] and IDx-DR[127] (which was approved by the FDA in 2021) all of which aim to improve existing healthcare services.

Furthermore, by advancing our understanding of breast cancer through machine learning-driven analyses, we can uncover novel insights into disease mechanisms, biomarker identification, and therapeutic targets[128]. This deeper understanding has the potential to fuel the development of innovative therapies and personalised treatment approaches, ultimately improving the quality of life for patients by offering more effective and targeted interventions[129].

Completing this thesis has opened up several opportunities for future work. Whilst our work has been completed to a high standard and with a successful result, it cannot be stressed enough how much continued effort into training and fine-tuning the aforementioned models will improve these models. Should there be more time dedicated to this, I believe that the hurdle of accuracy may be overcome such that the ML model can predict breast cancer subtypes with higher confidence and accuracy comparable to that of human specialists. An important idea that stemmed from this master's dissertation was packaging and deploying the trained prediction model into a web service/ app and integrating it into existing clinical workflows. This would enable accessibility of the software to both patients and clinicians, hereby streamlining the oncology pipeline. It would also save patients and doctors critical time and money that would otherwise be lost to manually identifying subtypes (which is also prone to human error). Whilst it is true that similar programmes have already been designed (Predict Breast[130], Enhanced Breast Cancer Prediction (EBCP)[131], Transpara[132], etc.), these are relatively new and still in developmental stages where they are yet to be implemented at scale. Unfortunately, in this case since it has proven difficult to combine classification of malignant subtypes and benign subtypes into one program, the model is not fit for integration into a web interface. Nevertheless, if there is continued research into combining the existing AI programs for both malignant and benign groups into one program, and the precision and recall is at a high enough level (greater than 90%), this would certainly render the program fit for web integration and subsequent incorporation into healthcare systems.

To conclude, the discussed work presents a pivotal first step towards harnessing machine learning to improve breast cancer research and care. By accelerating diagnosis, optimising treatment strategies, and advancing our understanding of the disease, we have the potential to redefine the existing cancer care pipeline, offering enhanced outcomes for both patients and clinicians.

5 Conclusion

This body of work has shown that our custom CNN holds great success in classifying malignant breast cancer into distinct subtypes. In fact, it outperforms the respected image recognition model EfficientNetB0 in the case of malignant breast cancer subtypes. Nevertheless, ResNet50 remains the top performer in image classification tasks for both benign and malignant breast cancer cases and should continue to be used in transfer learning for multiclass classification models such as this. Beyond good model performance, this work also highlights the increased demand for machine learning in cancer research and healthcare in general. We have shown that increased efforts into improving and incorporating existing AI models into healthcare systems can have an undeniable impact and make for an overall more efficient industry. Imploring AI into existing healthcare solutions is well within reach and if we consider how well these can perform, it is undoubtedly the next step in the evolution of healthcare.

6 Appendices

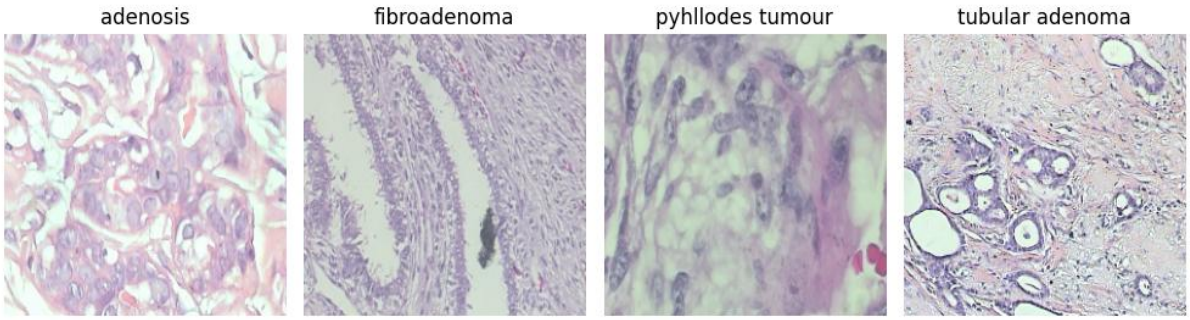


Figure 29: Example benign breast cancer images from each subtype

7 References

[1] Kaul, Vivek, Sarah Enslin, and Seth A. Gross. 2020. “History of Artificial Intelligence in Medicine.” Gastrointestinal Endoscopy 92 (4): 807–12. https://doi.org/10.1016/j.gie.2020.06.040.

[2] Jiang, F., Jiang, Y., Zhi, H., Dong, Y., Li, H., Ma, S., Wang, Y., Dong, Q., Shen, H., & Wang, Y. 2017. “Artificial Intelligence in healthcare: Past, present and future”. Stroke and Vascular Neurology. https://svn.bmj.com/content/2/4/230

[3] Tursunbayeva, A. and Renkema, M. 2023, “Artificial intelligence in health-care: implications for the job design of healthcare professionals”. Asia Pac J Hum Resour, 61: 845-887. https://doi.org/10.1111/1744-7941.12325

[4] Maghdid, H. S., Asaad, A. T., Ghafoor, K. Z., Sadiq, A. S., Mirjalili, S., & Khan, M. K. 2021. “Diagnosing COVID-19 pneumonia from X-ray and CT images using deep learning and transfer learning algorithms”. In Multimodal image exploitation and learning 2021 (Vol. 11734, pp. 99-110). SPIE.

[5] Lee, DonHee, and Seong No Yoon. 2021. "Application of Artificial Intelligence-Based Technologies in the Healthcare Industry: Opportunities and Challenges" International Journal of Environmental Research and Public Health 18, no. 1: 271. https://doi.org/10.3390/ijerph18010271

[6] Alowais, S.A., Alghamdi, S.S., Alsuhebany, N. et al. 2023. “Revolutionizing healthcare: the role of artificial intelligence in clinical practice”. BMC Med Educ 23, 689. https://doi.org/10.1186/s12909-023-04698-z

[7] Godet, I., & M. Gilkes, D. (2017). BRCA1 and BRCA2 mutations and treatment strategies for breast cancer. *Integrative Cancer Science and Therapeutics*, *4*(1). https://doi.org/10.15761/icst.1000228

[8] Fu, X., Tan, W., Song, Q., Pei, H., & Li, J. (2022). BRCA1 and Breast Cancer: Molecular Mechanisms and Therapeutic Strategies. *Frontiers in Cell and Developmental Biology*, *10*. https://doi.org/10.3389/fcell.2022.813457

[9] Wilkinson, L., & Gathani, T. (2021). Understanding breast cancer as a global health concern. *The British Journal of Radiology*, *95*(1130). https://doi.org/10.1259/bjr.20211033

[10] Barrios, C. H. (2022). Global challenges in breast cancer detection and treatment. *The Breast*, *62*(1). https://doi.org/10.1016/j.breast.2022.02.003

[11] Black, E., & Richmond, R. (2019). Improving early detection of breast cancer in sub-Saharan Africa: why mammography may not be the way forward. *Globalization and Health*, *15*(1). https://doi.org/10.1186/s12992-018-0446-6

[12] Moo, T.-A., Sanford, R., Dang, C., & Morrow, M. (2018). Overview of Breast Cancer Therapy. *PET Clinics*, *13*(3), 339–354. https://doi.org/10.1016/j.cpet.2018.02.006

[13] Wilkinson, L., & Gathani, T. (2021). Understanding Breast Cancer as a Global Health Concern. *The British Journal of Radiology*, *95*(1130). https://doi.org/10.1259/bjr.20211033

[14] Imran, M. T., Shafi, I., Ahmad, J., Butt, M. F. U., Villar, S. G., Villena, E. G., Khurshaid, T., & Ashraf, I. (2024). Virtual histopathology methods in medical imaging - a systematic review. *BMC Medical Imaging*, *24*(1). https://doi.org/10.1186/s12880-024-01498-9

[15] National Cancer Institute. 2021. "What Is Cancer?" National Cancer Institute. National Institutes of Health. 2021. https://www.cancer.gov/about-cancer/understanding/what-is-cancer.

[16] Gutschner, T., & Diederichs, S. 2012. "The hallmarks of cancer: a long non-coding RNA point of view". RNA biology, 9(6), 703–719. https://doi.org/10.4161/rna.20481

[17] Weigelt, B., Geyer, F. C., & Reis-Filho, J. S. 2010. "Histological types of breast cancer: how special are they?". Molecular oncology, 4(3), 192–208. https://doi.org/10.1016/j.molonc.2010.04.004

[18] Weigelt, B, HM Horlings, B Kreike, MM Hayes, M Hauptmann, LFA Wessels, D de Jong, MJ Van de Vijver, LJ Van't Veer, and JL Peterse. 2008. "Refinement of Breast Cancer Classification by Molecular Characterization of Histological Special Types." The Journal of Pathology 216 (2): 141–50. https://doi.org/10.1002/path.2407.

[19] Makki J. 2015. "Diversity of Breast Carcinoma: Histological Subtypes and Clinical Relevance". Clinical medicine insights. Pathology, 8, 23–31. https://doi.org/10.4137/CPath.S31563

[20] Łukasiewicz, Sergiusz. 2021. "Breast Cancer—Epidemiology, Risk Factors, Classification, Prognostic Markers, and Current Treatment Strategies—an Updated Review." Cancers 13 (17): 4287. https://doi.org/10.3390/cancers13174287.

[21] Vorobiof D, Sitas F, Vorobiof G. 2001. "Breast cancer incidence in South Africa. Journal of Clinical Oncology". (September 15 Supplement).; Vol 19, No. 18s: 125s - 127s

[22] World Health Organization. 2024. "Breast Cancer." World Health Organization. March 13, 2024. https://www.who.int/news-room/fact-sheets/detail/breast-cancer.

[23] Sharma, Rajesh. 2019. "Breast Cancer Incidence, Mortality and Mortality-To-Incidence Ratio (MIR) Are Associated with Human Development, 1990–2016:

Evidence from Global Burden of Disease Study 2016." Breast Cancer 26 (4): 428–45. https://doi.org/10.1007/s12282-018-00941-4.

[24] Arnold, Melina, Eileen Morgan, Harriet Rumgay, Allini Mafra, Deependra Singh, Mathieu Laversanne, Jerome Vignat, et al. 2022. "Current and Future Burden of Breast Cancer: Global Statistics for 2020 and 2040." The Breast 66 (66). https://doi.org/10.1016/j.breast.2022.08.010.

[25] Porter, Peggy. 2008. "'Westernizing' Women's Risks? Breast Cancer in Lower-Income Countries." New England Journal of Medicine 358 (3): 213–16. https://doi.org/10.1056/nejmp0708307.

[26] Duggan, Catherine, Allison Dvaladze, Anne F. Rositch, Ophira Ginsburg, Cheng-Har Yip, Susan Horton, Rolando Camacho Rodriguez, et al. 2020. "The Breast Health Global Initiative 2018 Global Summit on Improving Breast Healthcare through Resource-Stratified Phased Implementation: Methods and Overview." Cancer 126 (S10): 2339–52. https://doi.org/10.1002/cncr.32891.

[27] Ginsburg, Ophira, Freddie Bray, Michel P Coleman, Verna Vanderpuye, Alexandru Eniu, S Rani Kotha, Malabika Sarker, et al. 2017. "The Global Burden of Women's Cancers: A Grand Challenge in Global Health." The Lancet 389 (10071): 847–60. https://doi.org/10.1016/s0140-6736(16)31392-7.

[28] Bhushan, A., Gonsalves, A., & Menon, J. U. 2021. "Current State of Breast Cancer Diagnosis, Treatment, and Theranostics". Pharmaceutics, 13(5), 723. https://doi.org/10.3390/pharmaceutics13050723

[29] Lukong, Kiven Erique. 2017. "Understanding Breast Cancer – the Long and Winding Road." BBA Clinical 7 (June): 64–77. https://doi.org/10.1016/j.bbacli.2017.01.001.

[30] Holloway, C. M., Easson, A., Escallon, J., Leong, W. L., Quan, M. L., Reedjik, M., Wright, F. C., & McCready, D. R. 2010. "Technology as a force for improved diagnosis and treatment of breast disease". Canadian journal of surgery. Journal canadien de chirurgie, 53(4), 268–277.

[31] Badve S. 2018. "Predictive biomarkers and targeted therapies in breast cancer". Predictive Biomarkers in Oncology. 393–402. doi:10.1007/978-3-319-95228-4_35

[32] Targeted drug therapy [Internet]. [cited 2024 May 15]. Available from: https://www.cancer.org/cancer/managing-cancer/treatment-types/targeted-therapy.html

[33] Curigliano G. 2012. "New drugs for breast cancer subtypes: Targeting driver pathways to overcome resistance". Cancer Treatment Reviews. 38(4):303–10. doi: 10.1016/j.ctrv.2011.06.006

[34] Targeted cancer therapies south Africa: A.B.J. Inc [Internet]. 2023 [cited 2023 Sept 12]. Available from: https://www.oncology-sa.co.za/targeted-therapies/

[35] Chen X, Shachter RD, Kurian AW, Rubin DL. 2017. "Dynamic strategy for personalized medicine: An application to metastatic breast cancer". Journal of Biomedical Informatics. 68:50–7. doi: 10.1016/j.jbi.2017.02.012

[36] Curigliano G. 2012. "New drugs for breast cancer subtypes: Targeting driver pathways to overcome resistance". Cancer Treatment Reviews. 38(4):303–10. doi: 10.1016/j.ctrv.2011.06.006

[37] van den Bent MJ. 2010. "Interobserver variation of the histopathological diagnosis in clinical trials on glioma: A clinician's perspective". Acta Neuropathologica. 120(3):297–304. doi:10.1007/s00401-010-0725-7

[38] Guo L, Kong D, Liu J, Zhan L, Luo L, Zheng W, et al. 2023. "Breast cancer heterogeneity and its implication in personalized precision therapy". Experimental Hematology & Oncology. 12(1). doi:10.1186/s40164-022-00363-1

[39] Franco, E. F., Rana, P., Cruz, A., Calderón, V. V., Azevedo, V., Ramos, R. T. J., & Ghosh, P. 2021. "Performance Comparison of Deep Learning Autoencoders for Cancer Subtype Detection Using Multi-Omics Data". Cancers, 13(9), 2013. https://doi.org/10.3390/cancers13092013

[40] Takahashi, S., Takahashi, M., Tanaka, S., Takayanagi, S., Takami, H., Yamazawa, E., Nambu, S., Miyake, M., Satomi, K., Ichimura, K., Narita, Y., & Hamamoto, R. 2021. "A New Era of Neuro-Oncology Research Pioneered by Multi-Omics Analysis and Machine Learning". Biomolecules, 11(4), 565. https://doi.org/10.3390/biom11040565

[41] Liu, J., Xia, C., & Wang, G. 2020. "Multi-Omics Analysis in Initiation and Progression of Meningiomas: From Pathogenesis to Diagnosis". Frontiers in oncology, 10, 1491. https://doi.org/10.3389/fonc.2020.01491

[42] El Naqa, Issam, Aleksandra Karolak, Yi Luo, Les Folio, Ahmad A. Tarhini, Dana Rollison, and Katia Parodi. 2023. "Translation of AI into Oncology Clinical Practice." Oncogene 42 (September): 1–9. https://doi.org/10.1038/s41388-023-02826-z.

[43] Sun, D., M. Wang, and A. Li. 2019. "A Multimodal Deep Neural Network for Human Breast Cancer Prognosis Prediction by Integrating Multi-Dimensional Data." IEEE/ACM Transactions on Computational Biology and Bioinformatics 16 (3): 841–50. https://doi.org/10.1109/TCBB.2018.2806438.

[44] Zhao, Jianhui, Liying Xu, Jing Sun, Mingyang Song, Lijuan Wang, Shuai Yuan, Yingshuang Zhu, et al. 2023. "Global Trends in Incidence, Death, Burden and Risk Factors of Early-Onset Cancer from 1990 to 2019." BMJ Oncology 2 (1): e000049. https://doi.org/10.1136/bmjonc-2023-000049.

[45] Kavidopoulou, A. et al. 2022. “AI and Big Data for Drug Discovery”. In: Sakly, H., Yeom, K., Halabi, S., Said, M., Seekins, J., Tagina, M. (eds) Trends of Artificial Intelligence and Big Data for E-Health. Integrated Science, vol 9. Springer, Cham. https://doi.org/10.1007/978-3-031-11199-0_7

[46] Jiang, Peng, Sanju Sinha, Kenneth Aldape, Sridhar Hannenhalli, Cenk Sahinalp, and Eytan Ruppin. 2022. “Big Data in Basic and Translational Cancer Research.” Nature Reviews Cancer 22 (September): 1–15. https://doi.org/10.1038/s41568-022-00502-0.

[47] Blokker, Max, Philip C. de Witt Hamer, Pieter Wesseling, Marie Louise Groot, and Mitko Veta. 2022. “Fast Intraoperative Histology-Based Diagnosis of Gliomas with Third Harmonic Generation Microscopy and Deep Learning.” Scientific Reports 12 (1). https://doi.org/10.1038/s41598-022-15423-z.

[48] Goldenberg, S. Larry, Guy Nir, and Septimiu E. Salcudean. 2019. “A New Era: Artificial Intelligence and Machine Learning in Prostate Cancer.” Nature Reviews Urology 16 (7): 391–403. https://doi.org/10.1038/s41585-019-0193-3.

[49] Srinidhi, Chetan L., Ozan Ciga, and Anne L. Martel. 2021. “Deep Neural Network Models for Computational Histopathology: A Survey.” Medical Image Analysis 67 (January): 101813. https://doi.org/10.1016/j.media.2020.101813.

[50] Gu, Jiuxiang, Zhenhua Wang, Jason Kuen, Lianyang Ma, Amir Shahroudy, Bing Shuai, Ting Liu, et al. 2018. “Recent Advances in Convolutional Neural Networks.” Pattern Recognition 77 (May): 354–77. https://doi.org/10.1016/j.patcog.2017.10.013.

[51] Montesinos López, Osval Antonio, Abelardo Montesinos López, and Jose Crossa. 2022. “Convolutional Neural Networks.” Multivariate Statistical Machine Learning Methods for Genomic Prediction, 533–77. https://doi.org/10.1007/978-3-030-89010-0_13.

[52] Wang, Yingying, Yibin Li, Yong Song, and Xuewen Rong. 2020. "The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition" Applied Sciences 10, no. 5: 1897. https://doi.org/10.3390/app10051897

[53] Dabeer, Sumaiya, Maha Mohammed Khan, and Saiful Islam. 2019. “Cancer Diagnosis in Histopathological Image: CNN Based Approach.” Informatics in Medicine Unlocked 16: 100231. https://doi.org/10.1016/j.imu.2019.100231.

[54] Yu, Hualong, Houjuan Xie, Xibei Yang, Haitao Zou, and Shang Gao. 2021. “Online Sequential Extreme Learning Machine with the Increased Classes.” Computers & Electrical Engineering 90 (March): 107008. https://doi.org/10.1016/j.compeleceng.2021.107008.

[55] Xu, Siqiu, Xi Li, Chenchen Xie, Houpeng Chen, Cheng Chen, and Zhitang Song. 2021. “A High-Precision Implementation of the Sigmoid Activation Function for

Computing-In-Memory Architecture." Micromachines 12 (10): 1183–83. https://doi.org/10.3390/mi12101183.

[56] Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. 2018. "Activation functions: Comparison of trends in practice and research for deep learning". arXiv preprint arXiv:1811.03378.

[57] Gholamalinezhad, H., & Khosravi, H. 2020. "Pooling methods in deep neural networks, a review". arXiv preprint arXiv:2009.07485.

[58] Yamashita, R., Nishio, M., Do, R.K.G. *et al.* 2018. "Convolutional neural networks: an overview and application in radiology". Insights Imaging 9, 611–629. https://doi.org/10.1007/s13244-018-0639-9

[59] Milosevic, Nemanja. 2020. "Introduction to Convolutional Neural Networks". Apress EBooks. https://doi.org/10.1007/978-1-4842-5648-0.

[60] Zhong Liqun, Lingrui Li, and Ge Yang. 2022. "Characterizing Robustness of Deep Neural Networks in Semantic Segmentation of Fluorescence Microscopy Images." OPAL (Open@LaTrobe) (La Trobe University), July. https://doi.org/10.36227/techrxiv.20188742.v1.

[61] Li, Shutao, Weiwei Song, Leyuan Fang, Yushi Chen, Pedram Ghamisi, and Jón Atli Benediktsson. 2019. "Deep Learning for Hyperspectral Image Classification: An Overview." IEEE Transactions on Geoscience and Remote Sensing 57 (9): 6690–6709. https://doi.org/10.1109/TGRS.2019.2907932.

[62] Wang, Yingying, Yibin Li, Yong Song, and Xuewen Rong. 2020. "The Influence of the Activation Function in a Convolution Neural Network Model of Facial Expression Recognition" *Applied Sciences* 10, no. 5: 1897. https://doi.org/10.3390/app10051897

[63] Salehi, Ahmad Waleed, Shakir Khan, Gaurav Gupta, Bayan Ibrahimm Alabduallah, Abrar Almjally, Hadeel Alsolai, Tamanna Siddiqui, and Adel Mellit. 2023. "A Study of CNN and Transfer Learning in Medical Imaging: Advantages, Challenges, Future Scope" *Sustainability* 15, no. 7: 5930. https://doi.org/10.3390/su15075930

[64] Bahl, M., Barzilay, R., Yedidia, A. B., Locascio, N. J., Yu, L., & Lehman, C. D. 2018. "High-Risk Breast Lesions: A Machine Learning Model to Predict Pathologic Upgrade and Reduce Unnecessary Surgical Excision". Radiology, 286(3), 810–818. https://doi.org/10.1148/radiol.2017170549

[65] Iqbal, M. J., Javed, Z., Sadia, H., Qureshi, I. A., Irshad, A., Ahmed, R., Malik, K., Raza, S., Abbas, A., Pezzani, R., & Sharifi-Rad, J. 2021. "Clinical applications of artificial intelligence and machine learning in cancer diagnosis: looking into the future". Cancer cell international, 21(1), 270. https://doi.org/10.1186/s12935-021-01981-1

[66] Shen, Li, Laurie R. Margolies, Joseph H. Rothstein, Eugene Fluder, Russell McBride, and Weiva Sieh. 2019. “Deep Learning to Improve Breast Cancer Detection on Screening Mammography.” Scientific Reports 9 (1). https://doi.org/10.1038/s41598-019-48995-4.

[67] Kebede, Samuel Rahimeto, Fraol Gelana Waldamichael, Taye Girma Debelee, Muluberhan Aleme, Wubalem Bedane, Bethelhem Mezgebu, and Zelalem Chimdesa Merga. 2024. “Dual View Deep Learning for Enhanced Breast Cancer Screening Using Mammography.” Scientific Reports 14 (1): 3839. https://doi.org/10.1038/s41598-023-50797-8

[68] Alloqmani, Ahad, Yoosef B. Abushark, and Asif Irshad Khan. 2023. “Anomaly Detection of Breast Cancer Using Deep Learning.” Arabian Journal for Science and Engineering, June 1–26. https://doi.org/10.1007/s13369-023-07945-z.

[69] Lamin Juwara, Navpreet Arora, Mervyn Gornitsky, Paramita Saha-Chaudhuri, and Ana M Velly. 2020. “Identifying Predictive Factors for Neuropathic Pain after Breast Cancer Surgery Using Machine Learning.” International Journal of Medical Informatics 141 (September): 104170–70. https://doi.org/10.1016/j.ijmedinf.2020.104170.

[70] Bahl, Manisha, Regina Barzilay, Adam B. Yedidia, Nicholas J. Locascio, Lili Yu, and Constance D. Lehman. 2018. “High-Risk Breast Lesions: A Machine Learning Model to Predict Pathologic Upgrade and Reduce Unnecessary Surgical Excision.” Radiology 286 (3): 810–18. https://doi.org/10.1148/radiol.2017170549.

[71] Botlagunta, Mahendran, Madhavi Devi Botlagunta, Madhu Bala Myneni, D. Lakshmi, Anand Nayyar, Jaithra Sai Gullapalli, and Mohd Asif Shah. 2023. “Classification and Diagnostic Prediction of Breast Cancer Metastasis on Clinical Data Using Machine Learning Algorithms.” Scientific Reports 13 (1). https://doi.org/10.1038/s41598-023-27548-w.

[72] Liew, Xin Yu, Nazia Hameed, and Jeremie Clos. 2021. “An Investigation of XGBoost-Based Algorithm for Breast Cancer Classification.” Machine Learning with Applications, September, 100154. https://doi.org/10.1016/j.mlwa.2021.100154.

[73] Akinnuwesi, Boluwaji A., Babafemi O. Macaulay, and Benjamin S. Aribisala. 2020. “Breast Cancer Risk Assessment and Early Diagnosis Using Principal Component Analysis and Support Vector Machine Techniques.” Informatics in Medicine Unlocked 21 (January): 100459. https://doi.org/10.1016/j.imu.2020.100459.

[74] Wang, Yi, Eun Jung Choi, Younhee Choi, Hao Zhang, Gong Yong Jin, and Seok-Bum Ko. 2020. “Breast Cancer Classification in Automated Breast Ultrasound Using Multiview Convolutional Neural Network with Transfer Learning.” Ultrasound in Medicine & Biology 46 (5): 1119–32. https://doi.org/10.1016/j.ultrasmedbio.2020.01.001.

[75] Oyelade, Olaide N., and Absalom E. Ezugwu. 2021. “A Bioinspired Neural Architecture Search Based Convolutional Neural Network for Breast Cancer Detection Using Histopathology Images.” Scientific Reports 11 (1). https://doi.org/10.1038/s41598-021-98978-7.

[76] (Health, Center for Devices and Radiological. 2021. “Artificial Intelligence and Machine Learning (AI/ML)-Enabled Medical Devices.” FDA, September. https://www.fda.gov/medical-devices/software-medical-device-samd/artificial-intelligence-and-machine-learning-aiml-enabled-medical-devices.

[77] (NHS. 2023. “NHS England» Artificial Intelligence (AI) and Machine Learning.” Www.england.nhs.uk. June 14, 2023. https://www.england.nhs.uk/long-read/artificial-intelligence-ai-and-machine-learning/.

[78] Radu, Roxana. 2021. “Steering the Governance of Artificial Intelligence: National Strategies in Perspective.” Policy and Society 40 (2): 178–93. https://doi.org/10.1080/14494035.2021.1929728.

[79] Townsend, Beverley, Irvine Sihlahla, Meshandren Naidoo, Saloshni Naidoo, Dusty-Lee Donnelly, and Donrich Thaldar. 2023. “Mapping the Regulatory Landscape of AI in Healthcare in Africa.” Frontiers in Pharmacology 14 (August). https://doi.org/10.3389/fphar.2023.1214422.

[80] Phillips Foundation Team. 2021. “Philips Foundation Deploys AI Software in South Africa to Detect and Monitor COVID-19 Using Chest X-Rays.” Philips. March 25, 2021. https://www.philips-foundation.com/a-w/articles/CAD4COVID.html.

[81] Elsa Health. n.d. “Elsa Health | AI for Clinical Decision Support.” Www.elsa.health. Accessed April 19, 2024. https://www.elsa.health/.

[82] Babyl. n.d. “Services – Babyl.” Www.babyl.rw. Accessed April 19, 2024. https://www.babyl.rw/services/.

[83] Onu, Charles C., Jonathan Lebensold, William L. Hamilton, and Doina Precup. 2019. “Neural Transfer Learning for Cry-Based Diagnosis of Perinatal Asphyxia.” Interspeech 2019, September. https://doi.org/10.21437/interspeech.2019-2340.

[84] Thomasian, Nicole M., Carsten Eickhoff, and Eli Y. Adashi. 2021. “Advancing Health Equity with Artificial Intelligence.” Journal of Public Health Policy 42 (4): 602–11. https://doi.org/10.1057/s41271-021-00319-5.

[85] Barisoni, Laura, Kyle J. Lafata, Stephen M. Hewitt, Anant Madabhushi, and Ulysses G. J. Balis. 2020. “Digital Pathology and Computational Image Analysis in Nephropathology.” Nature Reviews Nephrology 16 (11): 669–85. https://doi.org/10.1038/s41581-020-0321-6.

[86] Ahn, J. S., Shin, S., Yang, S. A., Park, E. K., Kim, K. H., Cho, S. I., … & Kim, S. 2023. “Artificial Intelligence in Breast Cancer Diagnosis and Personalized Medicine”. Journal of Breast Cancer, 26(5), 405.

[87] Spanhol, F. A., Oliveira, L. S., Petitjean, C., & Heutte, L. (2016). A Dataset for Breast Cancer Histopathological Image Classification. IEEE Transactions on Biomedical Engineering, 63(7), 1455– 1462. https://doi.org/10.1109/tbme.2015.2496264

[88] Dissanayaka, S., Mudanayaka, O., Halloluwa, T., & De Silva, C. (2024). ImageLab: Simplifying Image Processing Exploration for Novices and Experts Alike. *arXiv preprint arXiv:2401.03157*.

[89] Orillo, J. W., Dela Cruz, J., Agapito, L., Satimbre, P. J., & Valenzuela, I. (2014). Identification of diseases in rice plant (Oryza sativa) using back propagation Artificial Neural Network. *2014 International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM)*. https://doi.org/10.1109/hnicem.2014.7016248

[90] Babu, R. G., Nedumaran, A., Manikandan, G., & Selvameena, R. (2022). Tensorflow: Machine learning using heterogeneous edge on distributed systems. *Deep Learning in Visual Computing and Signal Processing*, 71–90. https://doi.org/10.1201/9781003277224-4

[91] Silaparasetty, N. (2020). The Tensorflow Machine Learning Library. *Machine Learning Concepts with Python and the Jupyter Notebook Environment*, 149–171. https://doi.org/10.1007/978-1-4842-5967-2_8

[92] de Bruijne, M. (2016). Machine learning approaches in medical image analysis: From detection to diagnosis. *Medical Image Analysis*, *33*, 94–97. https://doi.org/10.1016/j.media.2016.06.032

[93] Building Blocks of Neural Networks. (2021). *Deep Learning for Physics Research*, 23–31. https://doi.org/10.1142/9789811237461_0003

[94] Ziegler, P., & Dittrich, K. R. (n.d.). Data Integration — problems, approaches, and perspectives. *Conceptual Modelling in Information Systems Engineering*, 39–58. https://doi.org/10.1007/978-3-540-72677-7_3

[95] Wang C, Chen MH, Schifano E, Wu J, Yan J. (2016). Statistical Methods and Computing for Big Data. Stat Interface 9, no. 4: 399-414. doi: 10.4310/SII. 2016.v9.n4.a1. PMID: 27695593; PMCID: PMC5041595.

[96] Dutta, S., Arunachalam, A., & Misailovic, S. (2022). To seed or not to seed? an empirical analysis of usage of seeds for testing in machine learning projects. *2022 IEEE Conference on Software Testing, Verification and Validation (ICST)*. https://doi.org/10.1109/icst53961.2022.00026

[97] Nanni, L., Fantozzi, C., & Lazzarini, N. (2015). Coupling different methods for overcoming the class imbalance problem. *Neurocomputing*, *158*, 48–61. https://doi.org/10.1016/j.neucom.2015.01.068

[98] Dong, Q., Gong, S., & Zhu, X. (2017). Class rectification hard mining for imbalanced deep learning. In Proceedings of the IEEE international conference on computer vision (pp. 1851-1860).

[99] Li, Y., & Vasconcelos, N. (2019). Repair: Removing representation bias by dataset resampling. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 9572-9581).

[100] A Study on the Relationships of Classifier Performance Metrics. Naeem Seliya, Taghi M. Khoshgoftaar, Jason Van Hulse. 2009, 21st IEEE International Conference on Tools with Artificial Intelligence, Newark, NJ, USA, pp. 59-66)

[101] Deep Convolutional Neural Networks Enable Discrimination of Heterogeneous Digital Pathology Images. Pegah Khosravi, Ehsan Kazemi, Marcin Imielinski, Olivier Elemento, Iman Hajirasouliha. 2018, EBioMedicine, pp. 317-328.)

[102] He, K., Zhang, X., Ren, S., & Sun, J. (2015, December 10). *Deep Residual Learning for Image Recognition*. ArXiv.org; arXiv. https://arxiv.org/abs/1512.03385

[103] Witten, I. H., Frank, E., & Hall, M. A. (2011). *Data Mining: Practical Machine Learning Tools and Techniques*. Elsevier.

[104] Pandey, K., & Patel, S. (2024). A theoretical perspective and experimental evaluation of the extensive analysis of loss functions in machine learning and Deep Learning. *2024 Asia Pacific Conference on Innovation in Technology (APCIT)*, *22*, 1–8. https://doi.org/10.1109/apcit62007.2024.10673427

[105] Jadon, S. (2020). A survey of loss functions for semantic segmentation. *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. https://doi.org/10.1109/cibcb48159.2020.9277638

[106] Kingma, D. P. (2014). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[107] R, M. T., Thakur, A., Gupta, M., Sinha, D. K., Mishra, K. K., Venkatesan, V. K., & Guluwadi, S. (2024). Transformative breast cancer diagnosis using CNNS with optimized ReduceLROnPlateau and early stopping enhancements. *International Journal of Computational Intelligence Systems*, *17*(1). https://doi.org/10.1007/s44196-023-00397-1

[108] Rasheed, Z., Ma, Y.-K., Ullah, I., Ghadi, Y. Y., Khan, M. Z., Khan, M. A., Abdusalomov, A., Alqahtani, F., & Shehata, A. M. (2023). Brain tumor classification

from MRI using image enhancement and convolutional neural network techniques. *Brain Sciences*, *13*(9), 1320. https://doi.org/10.3390/brainsci13091320

[109] Tharwat, A. (2021), "Classification assessment methods", *Applied Computing and Informatics*, Vol. 17 No. 1, pp. 168-192. https://doi.org/10.1016/j.aci.2018.08.003

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

[111] Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105-6114). PMLR.

[112] Sarwinda, D., Paradisa, R. H., Bustamam, A., & Anggia, P. (2021). Deep learning in image classification using residual network (ResNet) variants for detection of colorectal cancer. *Procedia Computer Science*, *179*, 423–431. https://doi.org/10.1016/j.procs.2021.01.025

[113] Shafiq, Muhammad, and Zhaoquan Gu. 2022. "Deep Residual Learning for Image Recognition: A Survey" *Applied Sciences* 12, no. 18: 8972. https://doi.org/10.3390/app12188972

[114] Mascarenhas, S., & Agarwal, M. (2021). A comparison between VGG16, VGG19 and Resnet50 architecture frameworks for Image Classification. *2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON)*. https://doi.org/10.1109/centcon52345.2021.9687944

[115] Goutham, V., Sameerunnisa, A., Babu, S., & Prakash, T. B. (2022). Brain tumor classification using EfficientNet-B0 model. *2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE)*, *9*, 2503–2509. https://doi.org/10.1109/icacite53722.2022.9823526

[116] Hoang, V.-T., & Jo, K.-H. (2021). Practical analysis on architecture of EfficientNet. *2021 14th International Conference on Human System Interaction (HSI)*, *269*, 1–4. https://doi.org/10.1109/hsi52170.2021.9538782

[117] Tan, M., & Le, Q. (2019, May). Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning* (pp. 6105-6114). PMLR.

[118] Spanhol, F. A., Oliveira, L. S., Petitjean, C., & Heutte, L. (2016). A Dataset for Breast Cancer Histopathological Image Classification. IEEE Transactions on Biomedical Engineering, 63(7), 1455– 1462. https://doi.org/10.1109/tbme.2015.2496264

[119] Yelne, S., Chaudhary, M., Dod, K., Sayyad, A., & Sharma, R. (2023). Harnessing the Power of AI: A Comprehensive Review of Its Impact and Challenges in Nursing

Science and Healthcare. Cureus, 15(11), e49252. https://doi.org/10.7759/cureus.49252

[120] Harnessing Artificial Intelligence for Health. (n.d.). Www.who.int. https://www.who.int/teams/digital-health-and-innovation/harnessing-artificial-intelligence-for-health

[121] Adewunmi Akingbola, Abiodun Adegbesan, Ojo, O., Jessica Urowoli Otumara, & Uthman Hassan Alao. (2024). Artificial Intelligence And Cancer Care in Africa. *Journal of Medicine Surgery and Public Health*, *3*, 100132–100132. https://doi.org/10.1016/j.glmedi.2024.100132

[122] Access CCG Data - NCI. (2022, June 26). Www.cancer.gov. https://www.cancer.gov/ccg/access-data

[123] Ahn, J. S., Shin, S., Yang, S. A., Park, E. K., Kim, K. H., Cho, S. I., Ock, C. Y., & Kim, S. (2023). Artificial Intelligence in Breast Cancer Diagnosis and Personalized Medicine. *Journal of breast cancer*, *26*(5), 405–435. https://doi.org/10.4048/jbc.2023.26.e45

[124] Jaganathan, D., Balasubramaniam, S., Sureshkumar, V., & Dhanasekaran, S. (2024). Revolutionizing Breast Cancer Diagnosis: A Concatenated Precision through Transfer Learning in Histopathological Data Analysis. *Diagnostics (Basel, Switzerland)*, *14*(4), 422. https://doi.org/10.3390/diagnostics14040422

[125] Zou, F. W., Tang, Y. F., Liu, C. Y., Ma, J. A., & Hu, C. H. (2020). Concordance Study Between IBM Watson for Oncology and Real Clinical Practice for Cervical Cancer Patients in China: A Retrospective Analysis. *Frontiers in genetics*, *11*, 200. https://doi.org/10.3389/fgene.2020.00200

[126] Dvijotham, K. (Dj), Winkens, J., Barsbey, M., Ghaisas, S., Stanforth, R., Pawlowski, N., Strachan, P., Ahmed, Z., Azizi, S., Bachrach, Y., Culp, L., Daswani, M., Freyberg, J., Kelly, C., Kiraly, A., Kohlberger, T., McKinney, S., Mustafa, B., Natarajan, V., & Geras, K. (2023). Enhancing the reliability and accuracy of AI-enabled diagnosis via complementarity-driven deferral to clinicians. *Nature Medicine*, *29*(7), 1814–1820. https://doi.org/10.1038/s41591-023-02437-x

[127] Grzybowski, A., & Brona, P. (2021). Analysis and comparison of two artificial intelligence diabetic retinopathy screening algorithms in a pilot study: IDx-DR and retinalyze. *Journal of clinical medicine*, *10*(11), 2352.

[128] Madani, M., Behzadi, M. M., & Nabavi, S. (2022). The Role of Deep Learning in Advancing Breast Cancer Detection Using Different Imaging Modalities: A Systematic Review. *Cancers*, *14*(21), 5334. https://doi.org/10.3390/cancers14215334

[129] Krzyszczyk, P., Acevedo, A., Davidoff, E. J., Timmins, L. M., Marrero-Berrios, I., Patel, M., White, C., Lowe, C., Sherba, J. J., Hartmanshenn, C., O'Neill, K. M., Balter, M. L., Fritz, Z. R., Androulakis, I. P., Schloss, R. S., & Yarmush, M. L. (2018). The growing role of precision and personalized medicine for cancer treatment. *Technology*, *6*(3-4), 79–100. https://doi.org/10.1142/S2339547818300020

[130] National Health Service (NHS). (n.d.). Breast cancer risk assessment. Predict: Breast Cancer Risk Prediction Tool. https://breast.predict.nhs.uk/

[131] MyEBCD Mammo. (n.d.). https://myebcdmammo.com/

[132] ScreenPoint Medical. (n.d.). Fusion AI. https://screenpoint-medical.com/fusion-ai/

```
import matplotlib.pyplot as plt
import numpy as np
import PIL
import tensorflow as tf
import pathlib
from PIL import Image
from pathlib import Path
import random
import cleanvision as cv
from cleanvision.imagelab import Imagelab
import os
import seaborn as sns
import pandas as pd

import sklearn as skl
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.utils import resample
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten, Conv2D,
MaxPooling2D, Dropout, BatchNormalization, GlobalAveragePooling2D,
Rescaling
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model
from tensorflow.keras.utils import image_dataset_from_directory as
tfds
from tensorflow.keras.callbacks import EarlyStopping,
ReduceLROnPlateau
from tensorflow.keras.metrics import AUC

from typing import Tuple
```

```
c:\Users\solyl\OneDrive - University of Cape Town\Masters\Code\.venv\
Lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found.
Please update jupyter and ipywidgets. See
https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
```

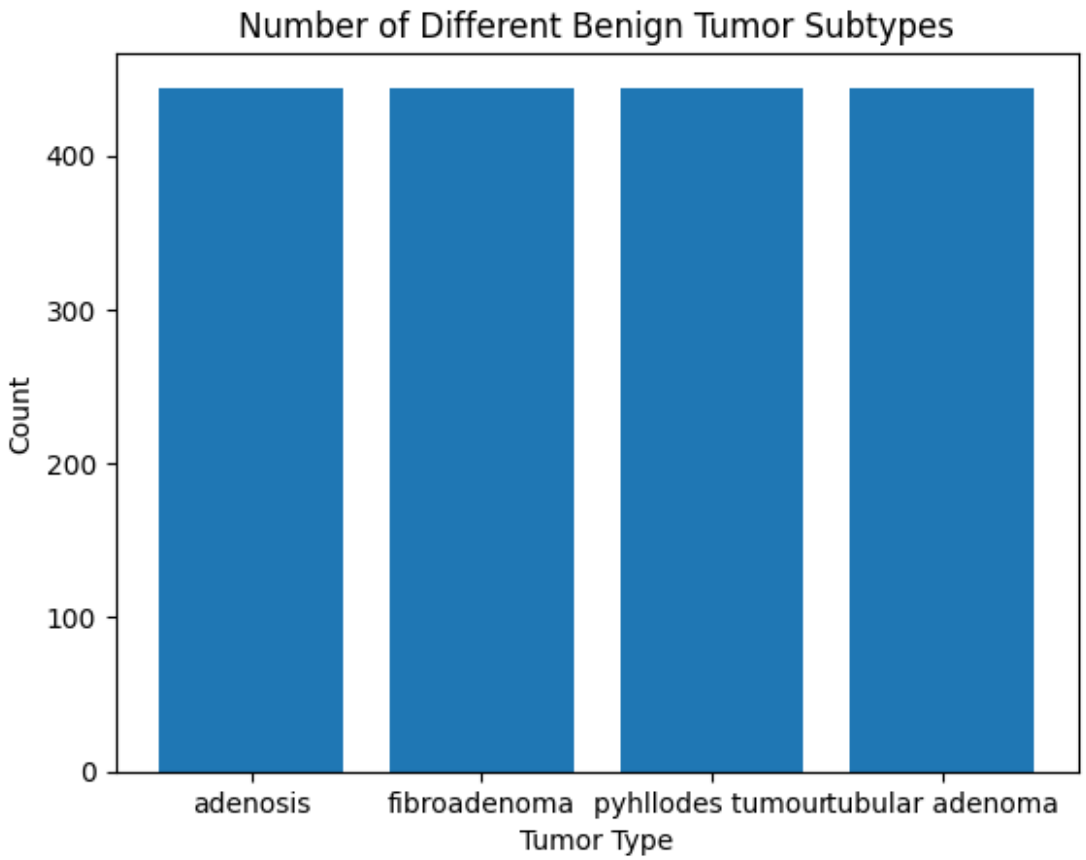
```
#loading the dataset - looking at only BENIGN
# Convert the string path to a Path object
data_dir = pathlib.Path("C:/Users/solyl/OneDrive - University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/benign/SOB")
```

```
# Use glob to count the number of .png files
image_count = len(list(data_dir.glob('**/*.png')))
print(image_count)

adenosis_dir = pathlib.Path("C:/Users/solyl/OneDrive - University of
Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/benign/SOB/adenosis")
adenosis_count = len(list(adenosis_dir.glob('**/*.png')))
print(adenosis_count)

fibroadenoma_dir = pathlib.Path("C:/Users/solyl/OneDrive - University
of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/benign/SOB/fibroadenoma")
fibroadenoma_count = len(list(fibroadenoma_dir.glob('**/*.png')))
print(fibroadenoma_count)

phyllodes_tumor_dir = pathlib.Path("C:/Users/solyl/OneDrive -
University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/benign/SOB/phyllodes_tumor")
phyllodes_tumor_count =
len(list(phyllodes_tumor_dir.glob('**/*.png')))
print(phyllodes_tumor_count)

tubular_adenoma_dir = pathlib.Path("C:/Users/solyl/OneDrive -
University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/benign/SOB/tubular_adenoma")
tubular_adenoma_count =
len(list(tubular_adenoma_dir.glob('**/*.png')))
print(tubular_adenoma_count)
```

```
1776
444
444
444
444
```

```
x = ["adenosis", "fibroadenoma", "pyhllodes tumour","tubular adenoma"]
y = [adenosis_count, fibroadenoma_count, phyllodes_tumor_count,
tubular_adenoma_count]
plt.bar(x,y)
plt.title('Number of Different Benign Tumor Subtypes')
plt.xlabel('Tumor Type')
plt.ylabel('Count')
```

```
Text(0, 0.5, 'Count')
```



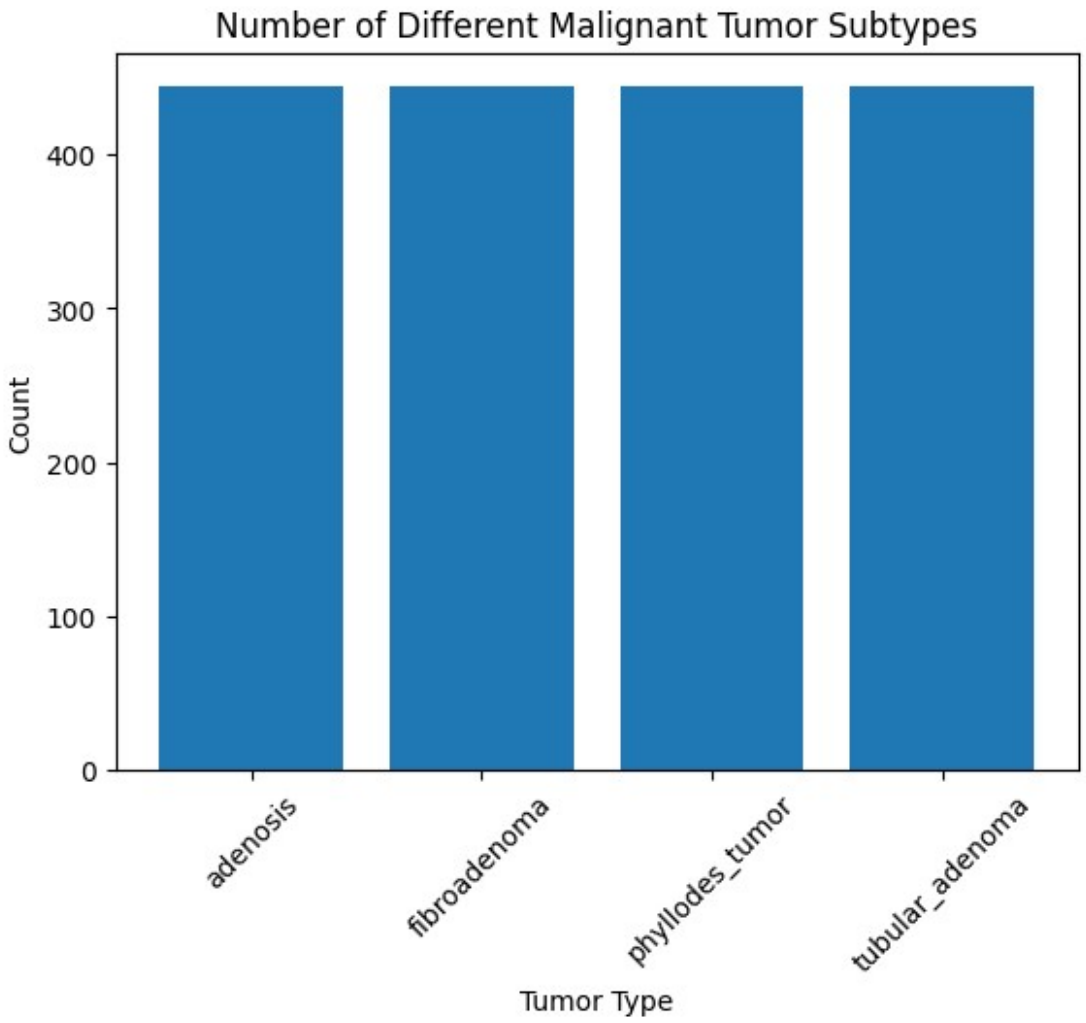

```
# Initialize an empty dictionary to store the file counts for each
folder.
class_counts = {}

# Path to the main directory containing subfolders.
main_dir = data_dir
min_images = 444

# Loop through each folder in the main directory.
for class_name in os.listdir(main_dir):
    class_dir = os.path.join(main_dir, class_name)

    # Check if the item is a directory (to exclude files in the
main_dir).
    if os.path.isdir(class_dir):
        # Initialize a file count for this folder.
        file_count = 0
        images = []

        # Walk through all subfolders in the class_dir to count files
and collect file paths.
        for root, dirs, files in os.walk(class_dir):
            for file in files:
```

```
                # Ensure we're only collecting files (and filtering
images, if needed)
                file_path = os.path.join(root, file)
                if os.path.isfile(file_path):  # Make sure it's a file
                    images.append(file_path)
                    file_count += 1

        # Store the count in the dictionary with the class_name as the
key.
        class_counts[class_name] = file_count

        # Debugging output to ensure we counted the correct number of
files
        print(f"{class_name} has {file_count} images.")

        # Now, delete images if there are more than min_images in this
class
        if file_count > min_images:
            excess_images = file_count - min_images
            print(f"{class_name} has {excess_images} excess images.
Preparing to delete...")

            # Randomly select images to delete (ensure we don't delete
more than we have)
            if excess_images > 0 and excess_images <= len(images):
                images_to_delete = random.sample(images,
excess_images)

                # Delete the selected images
                for image_path in images_to_delete:
                    print(f"Attempting to delete: {image_path}")  #
Debugging statement
                    try:
                        os.remove(image_path)
                        print(f"Deleted {image_path} from
{class_name}")
                    except Exception as e:
                        print(f"Error deleting {image_path}: {e}")
            else:
                print(f"Skipping deletion for {class_name}. No excess
images to delete.")
        else:
            print(f"No excess images to delete for {class_name}.")
    else:
        print(f"Skipping {class_name}, as it is not a directory.")

# Output the dictionary with counts after deletions
print("Final class counts after deletion:")
print(class_counts)
```

```
adenosis has 444 images.
No excess images to delete for adenosis.
fibroadenoma has 444 images.
No excess images to delete for fibroadenoma.
phyllodes_tumor has 444 images.
No excess images to delete for phyllodes_tumor.
tubular_adenoma has 444 images.
No excess images to delete for tubular_adenoma.
Final class counts after deletion:
{'adenosis': 444, 'fibroadenoma': 444, 'phyllodes_tumor': 444,
'tubular_adenoma': 444}
```

```
names = list(class_counts.keys())
counts = list(class_counts.values())


plt.bar(names, counts)
plt.title('Number of Different Malignant Tumor Subtypes')
plt.xlabel('Tumor Type')
plt.ylabel('Count')
plt.xticks(rotation=45)
```

```
([0, 1, 2, 3],
 [Text(0, 0, 'adenosis'),
  Text(1, 0, 'fibroadenoma'),
  Text(2, 0, 'phyllodes_tumor'),
  Text(3, 0, 'tubular_adenoma')])
```

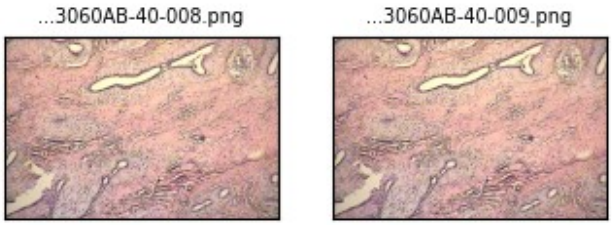



```
# Image Validation- using Imagelab from cleanvision to detect issues
in the image dataset
# ADENOSIS

adenosis_images = list(adenosis_dir.glob('**/*.png'))

adenosis_filepaths = [str(path) for path in adenosis_images]  #
Convert WindowsPath objects to strings

# Applying ImageLab
imagelab = Imagelab(filepaths=adenosis_filepaths)
imagelab.find_issues()
imagelab.report()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...
```

```
100%|██████████| 444/444 [00:09<00:00, 45.99it/s]
100%|██████████| 444/444 [00:07<00:00, 61.82it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | dark             |            0 |
|  1 | light            |            0 |
|  2 | odd_aspect_ratio |            0 |
|  3 | low_information  |            0 |
|  4 | blurry           |            0 |
|  5 | grayscale        |            0 |
|  6 | odd_size         |            0 |
|  7 | exact_duplicates |            0 |
|  8 | near_duplicates  |            0 |


#Image Validation
#using Imagelab from cleanvision to detect issues in the image dataset
#FIBROADENOMA

fibroadenoma_images = list(fibroadenoma_dir.glob('**/*.png'))

fibroadenoma_filepaths = [str(path) for path in fibroadenoma_images]
# Convert WindowsPath objects to strings

# Pass the list of string file paths to Imagelab
imagelab = Imagelab(filepaths=fibroadenoma_filepaths)
imagelab.find_issues()
imagelab.report()

Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 444/444 [00:17<00:00, 24.92it/s]
100%|██████████| 444/444 [00:08<00:00, 50.43it/s]

Issue checks completed. 2 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | near_duplicates  |            2 |
|  1 | dark             |            0 |
|  2 | light            |            0 |
|  3 | odd_aspect_ratio |            0 |
```

```
|  4 | low_information  |            0 |
|  5 | blurry           |            0 |
|  6 | grayscale        |            0 |
|  7 | odd_size         |            0 |
|  8 | exact_duplicates |            0 |

------------------ near_duplicates images ------------------

Number of examples with this issue: 2
Examples representing most severe instances of this issue:

Set: 0
```




```
# Filter images flagged as exact or near duplicates
duplicated_images = imagelab.issues[
    imagelab.issues[['is_exact_duplicates_issue',
'is_near_duplicates_issue']].any(axis=1)
].reset_index().rename(columns={'index': 'filename'})

# Display the filtered DataFrame to check which images are duplicates
print(duplicated_images.head())
```

```
                                            filename
odd_size_score  \
0  C:\Users\solyl\OneDrive - University of Cape T...             1.0

1  C:\Users\solyl\OneDrive - University of Cape T...             1.0


   is_odd_size_issue  odd_aspect_ratio_score
is_odd_aspect_ratio_issue  \
0              False                0.657143
False
1              False                0.657143
False

   low_information_score  is_low_information_issue  light_score  \
0               0.881227                     False     0.651887
1               0.881425                     False     0.652634

   is_light_issue  grayscale_score  is_grayscale_issue  dark_score  \
0           False                1               False    0.988192
```

```
1           False                 1               False    0.988118

   is_dark_issue  blurry_score  is_blurry_issue
exact_duplicates_score  \
0          False      0.458219            False
1.0
1          False      0.457516            False
1.0

   is_exact_duplicates_issue  near_duplicates_score
is_near_duplicates_issue
0                      False                   0.5
True
1                      False                   0.5
True
```

```
# Convert the filenames to a set for fast lookup
duplicate_files = set(duplicated_images['filename'])

# Remove duplicate images from the original filepaths list
cleaned_fibroadenoma_filepaths = [f for f in fibroadenoma_filepaths if
f not in duplicate_files]

# Display the number of files before and after removing duplicates
print(f"Original number of files: {len(fibroadenoma_filepaths)}")
print(f"Number of files after removing duplicates:
{len(cleaned_fibroadenoma_filepaths)}")
```

```
Original number of files: 444
Number of files after removing duplicates: 442
```

```
# Validate again
imagelab = Imagelab(filepaths=cleaned_fibroadenoma_filepaths)
imagelab.find_issues()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 442/442 [00:16<00:00, 26.76it/s]
100%|██████████| 442/442 [00:12<00:00, 34.41it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
```

```
#Image Validation
#using Imagelab from cleanvision to detect issues in the image dataset
#PHYLLODES TUMOUR

phyllodes_tumor_images = list(phyllodes_tumor_dir.glob('**/*.png'))
```

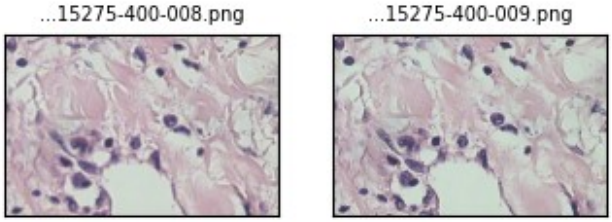
```
phyllodes_tumor_filepaths = [str(path) for path in
phyllodes_tumor_images]  # Convert WindowsPath objects to strings

# Pass the list of string file paths to Imagelab
imagelab = Imagelab(filepaths=phyllodes_tumor_filepaths)
imagelab.find_issues()
imagelab.report()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 444/444 [00:16<00:00, 27.58it/s]
100%|██████████| 444/444 [00:11<00:00, 37.50it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | dark             |            0 |
|  1 | light            |            0 |
|  2 | odd_aspect_ratio |            0 |
|  3 | low_information  |            0 |
|  4 | blurry           |            0 |
|  5 | grayscale        |            0 |
|  6 | odd_size         |            0 |
|  7 | exact_duplicates |            0 |
|  8 | near_duplicates  |            0 |
```

```
#Image Validation
#using Imagelab from cleanvision to detect issues in the image dataset
#TUBULAR ADENOMA

tubular_adenoma_images = list(tubular_adenoma_dir.glob('**/*.png'))

tubular_adenoma_filepaths = [str(path) for path in
tubular_adenoma_images]  # Convert WindowsPath objects to strings

# Pass the list of string file paths to Imagelab
imagelab = Imagelab(filepaths=tubular_adenoma_filepaths)
imagelab.find_issues()
imagelab.report()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...
```

```
100%|██████████| 444/444 [00:18<00:00, 24.29it/s]
100%|██████████| 444/444 [00:21<00:00, 20.34it/s]

Issue checks completed. 2 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | near_duplicates  |            2 |
|  1 | dark             |            0 |
|  2 | light            |            0 |
|  3 | odd_aspect_ratio |            0 |
|  4 | low_information  |            0 |
|  5 | blurry           |            0 |
|  6 | grayscale        |            0 |
|  7 | odd_size         |            0 |
|  8 | exact_duplicates |            0 |

------------------ near_duplicates images ------------------

Number of examples with this issue: 2
Examples representing most severe instances of this issue:

Set: 0
```




```
# Filter images flagged as exact or near duplicates
duplicated_images = imagelab.issues[
    imagelab.issues[['is_exact_duplicates_issue',
'is_near_duplicates_issue']].any(axis=1)
].reset_index().rename(columns={'index': 'filename'})

# Display the filtered DataFrame to check which images are duplicates
print(duplicated_images.head())

                                                filename
odd_size_score  \
0  C:\Users\solyl\OneDrive - University of Cape T...             1.0

1  C:\Users\solyl\OneDrive - University of Cape T...             1.0
```

```
   is_odd_size_issue  odd_aspect_ratio_score
is_odd_aspect_ratio_issue  \
0              False                0.657143
False
1              False                0.657143
False

   low_information_score  is_low_information_issue  light_score  \
0               0.808297                     False     0.537127
1               0.806036                     False     0.567823

   is_light_issue  grayscale_score  is_grayscale_issue  dark_score  \
0           False                1               False    0.882769
1           False                1               False    0.876015

   is_dark_issue  blurry_score  is_blurry_issue
exact_duplicates_score  \
0          False      0.507765            False
1.0
1          False      0.508253            False
1.0

   is_exact_duplicates_issue  near_duplicates_score
is_near_duplicates_issue
0                      False                   0.5
True
1                      False                   0.5
True
```

```
# Convert the filenames to a set for fast lookup
duplicate_files = set(duplicated_images['filename'])

# Remove duplicate images from the original filepaths list
cleaned_tubular_adenoma_filepaths = [f for f in
tubular_adenoma_filepaths if f not in duplicate_files]

# Display the number of files before and after removing duplicates
print(f"Original number of files: {len(tubular_adenoma_filepaths)}")
print(f"Number of files after removing duplicates:
{len(cleaned_tubular_adenoma_filepaths)}")
```

```
Original number of files: 444
Number of files after removing duplicates: 442
```

```
#Validate again
imagelab = Imagelab(filepaths=cleaned_tubular_adenoma_filepaths)
imagelab.find_issues()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...
```

```
100%|██████████| 442/442 [00:20<00:00, 21.58it/s]
100%|██████████| 442/442 [00:13<00:00, 32.17it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
```

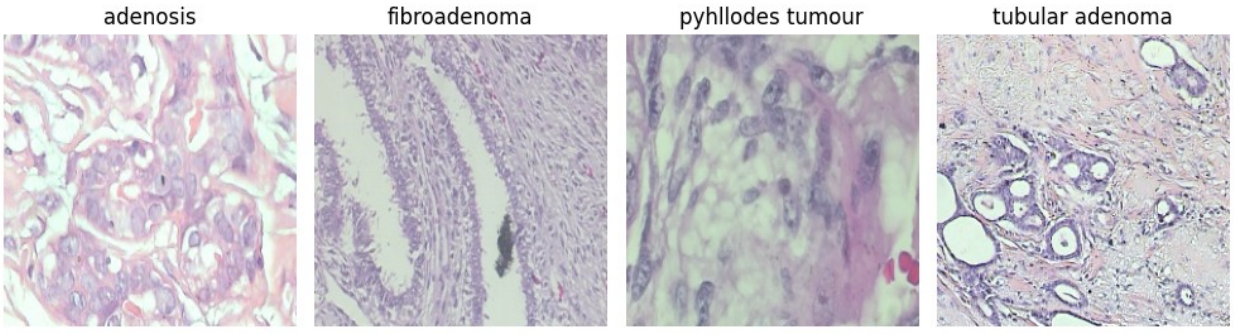
```python
batch_size = 28
img_height = 224
img_width = 224

# Load the entire dataset from the directory
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    data_dir,
    image_size=(img_height, img_width),
    batch_size=batch_size,
    shuffle=True
)

# Extract images and labels from the dataset
image_list = []
label_list = []

for images, labels in dataset:
    image_list.append(images.numpy())
    label_list.append(labels.numpy())

# Convert lists to NumPy arrays
X = np.concatenate(image_list)
y = np.concatenate(label_list)

# Split data into train (70%), validation (15%), and test (15%)
X_train, X_temp, y_train, y_temp = train_test_split(X, y,
test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp,
test_size=0.5, random_state=42)

# Create TensorFlow datasets for train, validation, and test sets
train_ds = tf.data.Dataset.from_tensor_slices((X_train,
y_train)).batch(batch_size)
validation_ds = tf.data.Dataset.from_tensor_slices((X_val,
y_val)).batch(batch_size)
test_ds = tf.data.Dataset.from_tensor_slices((X_test,
y_test)).batch(batch_size)

# Now you have train_dataset, validation_dataset, and test_dataset
```

```
Found 1776 files belonging to 4 classes.
```

```
label_mapping = {0: 'adenosis', 1: 'fibroadenoma', 2: 'pyhllodes
tumour', 3: 'tubular adenoma'}
labels = [0, 1, 2,3]  # Numeric labels
class_names = [label_mapping[label] for label in np.unique(labels)]
print("Class names:", class_names)
```

```
Class names: ['adenosis', 'fibroadenoma', 'pyhllodes tumour', 'tubular
adenoma']
```

```
# Extract true labels
y_true = np.concatenate([y.numpy() for x, y in validation_ds], axis=0)

# Print first 9 images and labels from the training set
unique_classes = np.unique(y)

plt.figure(figsize=(10, 10))

# Iterate over each class
for i, class_label in enumerate(unique_classes):
        # Find the first image in the dataset that corresponds to this
class
        class_index = np.where(y == class_label)[0][0]
        image = X[class_index]

        # Plot the image
        plt.subplot(1, len(unique_classes), i + 1)
        plt.imshow(image.astype('uint8'))
        plt.title(class_names[i])
        plt.axis('off')

plt.tight_layout()
plt.show()
```




```
for image_batch, labels_batch in train_ds:
  print(image_batch.shape)
  print(labels_batch.shape)
  break
```

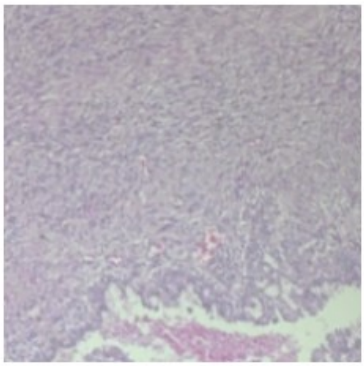
```
(28, 224, 224, 3)
(28,)
```

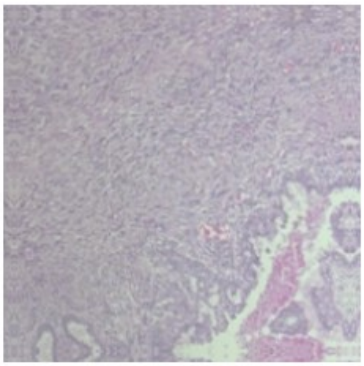
```
AUTOTUNE = tf.data.AUTOTUNE

train_ds =
train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
validation_ds = validation_ds.cache().prefetch(buffer_size=AUTOTUNE)
test_ds = test_ds.cache().prefetch(buffer_size=AUTOTUNE)

# Normalisation Layer
normalization_layer = layers.Rescaling(1./255)

normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
# Pixel values are now in `[0,1]`.
print(np.min(first_image), np.max(first_image))
```

```
0.2395172 1.0
```

```
# Data Augmentation
data_augmentation = keras.Sequential(
  [
    layers.RandomFlip("horizontal",
                      input_shape=(img_height,
                                   img_width,
                                   3)),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
  ]
)# View some of these
plt.figure(figsize=(10, 10))
for images, _ in train_ds.take(1):
  for i in range(3):
    augmented_images = data_augmentation(images)
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(augmented_images[0].numpy().astype("uint8"))
    plt.axis("off")
```

```
c:\Users\solyl\OneDrive - University of Cape Town\Masters\Code\.venv\
Lib\site-packages\keras\src\layers\preprocessing\tf_data_layer.py:18:
UserWarning: Do not pass an `input_shape`/`input_dim` argument to a
layer. When using Sequential models, prefer using an `Input(shape)`
object as the first layer in the model instead.
  super().__init__(**kwargs)
```





```
custom_model = tf.keras.Sequential([
    layers.Input(shape=(224, 224, 3)),
    # Data augmentation
    data_augmentation,
    # Convolutional block I
    layers.Rescaling(1./255),
    layers.BatchNormalization(),
    layers.Conv2D(32, 3, activation="relu"),
    layers.MaxPooling2D(),
    # Convolutional block II
    layers.Conv2D(64, 3, activation="relu"),
    layers.MaxPooling2D(),
    # Convolutional block III
    layers.Conv2D(128, 3, activation="relu"),
    layers.MaxPooling2D(),
    # Fully connected layers
    layers.GlobalAveragePooling2D(),
    layers.Dropout(0.4),
    layers.Dense(256, activation="relu"),
    layers.Dropout(0.3),
    layers.Dense(32, activation="relu"),
    layers.Dropout(0.2),
    layers.Dense(4, activation="softmax")
], name="CustomCNN")
custom_model.summary()
```

```
Model: "CustomCNN"
```

```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━
━━┓
┃ Layer (type)                         ┃ Output Shape                ┃
Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━
━━┩
│ sequential (Sequential)              │ (None, 224, 224, 3)         │
0 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────
──┘
```

```
| rescaling_1 (Rescaling)          | (None, 224, 224, 3)    |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| batch_normalization              | (None, 224, 224, 3)    |
12 |
| (BatchNormalization)             |                        |
|
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| conv2d (Conv2D)                  | (None, 222, 222, 32)   |
896 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| max_pooling2d (MaxPooling2D)     | (None, 111, 111, 32)   |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| conv2d_1 (Conv2D)                | (None, 109, 109, 64)   |
18,496 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| max_pooling2d_1 (MaxPooling2D)   | (None, 54, 54, 64)     |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| conv2d_2 (Conv2D)                | (None, 52, 52, 128)    |
73,856 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| max_pooling2d_2 (MaxPooling2D)   | (None, 26, 26, 128)    |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| global_average_pooling2d         | (None, 128)            |
0 |
| (GlobalAveragePooling2D)         |                        |
|
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| dropout (Dropout)                | (None, 128)            |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| dense (Dense)                    | (None, 256)            |
33,024 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| dropout_1 (Dropout)              | (None, 256)            |
```

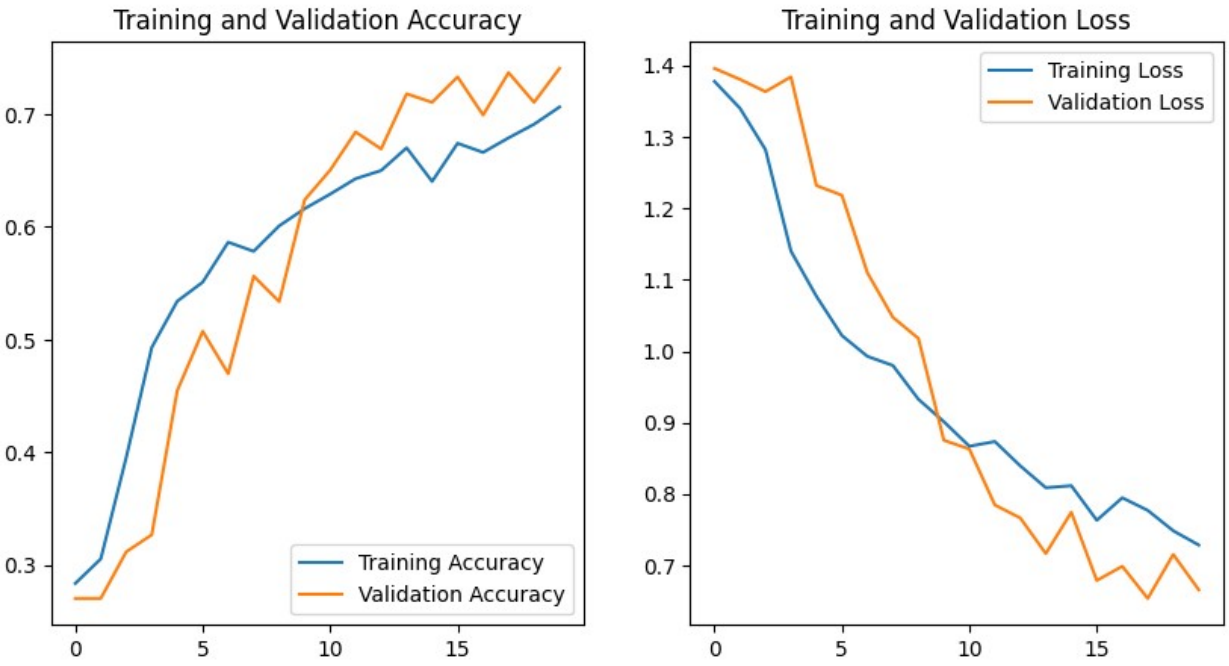
```
0 │
├─────────────────────────────────────┼─────────────────────────────┼──────────
──────┤
│ dense_1 (Dense)                     │ (None, 32)                  │
8,224 │
├─────────────────────────────────────┼─────────────────────────────┼──────────
──────┤
│ dropout_2 (Dropout)                 │ (None, 32)                  │
0 │
├─────────────────────────────────────┼─────────────────────────────┼──────────
──────┤
│ dense_2 (Dense)                     │ (None, 4)                   │
132 │
└─────────────────────────────────────┴─────────────────────────────┴──────────
──────┘

 Total params: 134,640 (525.94 KB)

 Trainable params: 134,634 (525.91 KB)

 Non-trainable params: 6 (24.00 B)
```

```
# Compile the model
custom_model.compile(optimizer=Adam(learning_rate=0.001),
              loss='sparse_categorical_crossentropy',  # Best to use
sparse categorical crossentropy for multi-class classification
              metrics=["accuracy"])

early_stopping = EarlyStopping(min_delta=1e-4, patience=5,
verbose=1,restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(factor=0.5, patience=4, verbose=1)

# Fit the model
epochs=20
history = custom_model.fit(train_ds, validation_data=validation_ds,
epochs=epochs, callbacks = [early_stopping, reduce_lr])
```

```
Epoch 1/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 80s 1s/step - accuracy: 0.2693 - loss:
1.3943 - val_accuracy: 0.2707 - val_loss: 1.3960 - learning_rate:
0.0010
Epoch 2/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 61s 1s/step - accuracy: 0.2860 - loss:
1.3572 - val_accuracy: 0.2707 - val_loss: 1.3806 - learning_rate:
0.0010
Epoch 3/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 65s 1s/step - accuracy: 0.3869 - loss:
1.3002 - val_accuracy: 0.3120 - val_loss: 1.3634 - learning_rate:
0.0010
Epoch 4/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 60s 1s/step - accuracy: 0.4736 - loss:
```

```
1.1655 - val_accuracy: 0.3271 - val_loss: 1.3842 - learning_rate:
0.0010
Epoch 5/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 56s 1s/step - accuracy: 0.5345 - loss:
1.0722 - val_accuracy: 0.4549 - val_loss: 1.2320 - learning_rate:
0.0010
Epoch 6/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 60s 1s/step - accuracy: 0.5312 - loss:
1.0697 - val_accuracy: 0.5075 - val_loss: 1.2183 - learning_rate:
0.0010
Epoch 7/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 66s 1s/step - accuracy: 0.5892 - loss:
0.9803 - val_accuracy: 0.4699 - val_loss: 1.1098 - learning_rate:
0.0010
Epoch 8/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 54s 1s/step - accuracy: 0.5810 - loss:
0.9580 - val_accuracy: 0.5564 - val_loss: 1.0475 - learning_rate:
0.0010
Epoch 9/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 55s 1s/step - accuracy: 0.6133 - loss:
0.9321 - val_accuracy: 0.5338 - val_loss: 1.0179 - learning_rate:
0.0010
Epoch 10/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 61s 1s/step - accuracy: 0.6057 - loss:
0.9192 - val_accuracy: 0.6241 - val_loss: 0.8753 - learning_rate:
0.0010
Epoch 11/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 67s 1s/step - accuracy: 0.6221 - loss:
0.8758 - val_accuracy: 0.6504 - val_loss: 0.8630 - learning_rate:
0.0010
Epoch 12/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 64s 1s/step - accuracy: 0.6353 - loss:
0.8975 - val_accuracy: 0.6842 - val_loss: 0.7848 - learning_rate:
0.0010
Epoch 13/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 68s 2s/step - accuracy: 0.6756 - loss:
0.7763 - val_accuracy: 0.6692 - val_loss: 0.7664 - learning_rate:
0.0010
Epoch 14/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 69s 2s/step - accuracy: 0.6676 - loss:
0.8050 - val_accuracy: 0.7180 - val_loss: 0.7167 - learning_rate:
0.0010
Epoch 15/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 70s 2s/step - accuracy: 0.6372 - loss:
0.8149 - val_accuracy: 0.7105 - val_loss: 0.7745 - learning_rate:
0.0010
Epoch 16/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 71s 2s/step - accuracy: 0.6712 - loss:
0.7970 - val_accuracy: 0.7331 - val_loss: 0.6790 - learning_rate:
```

```
0.0010
Epoch 17/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 71s 2s/step - accuracy: 0.6474 - loss:
0.8280 - val_accuracy: 0.6992 - val_loss: 0.6989 - learning_rate:
0.0010
Epoch 18/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 71s 2s/step - accuracy: 0.6841 - loss:
0.7547 - val_accuracy: 0.7368 - val_loss: 0.6539 - learning_rate:
0.0010
Epoch 19/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 70s 2s/step - accuracy: 0.7014 - loss:
0.7421 - val_accuracy: 0.7105 - val_loss: 0.7154 - learning_rate:
0.0010
Epoch 20/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 71s 2s/step - accuracy: 0.6872 - loss:
0.7610 - val_accuracy: 0.7406 - val_loss: 0.6660 - learning_rate:
0.0010
Restoring model weights from the end of the best epoch: 18.

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(len(history.history['accuracy']))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

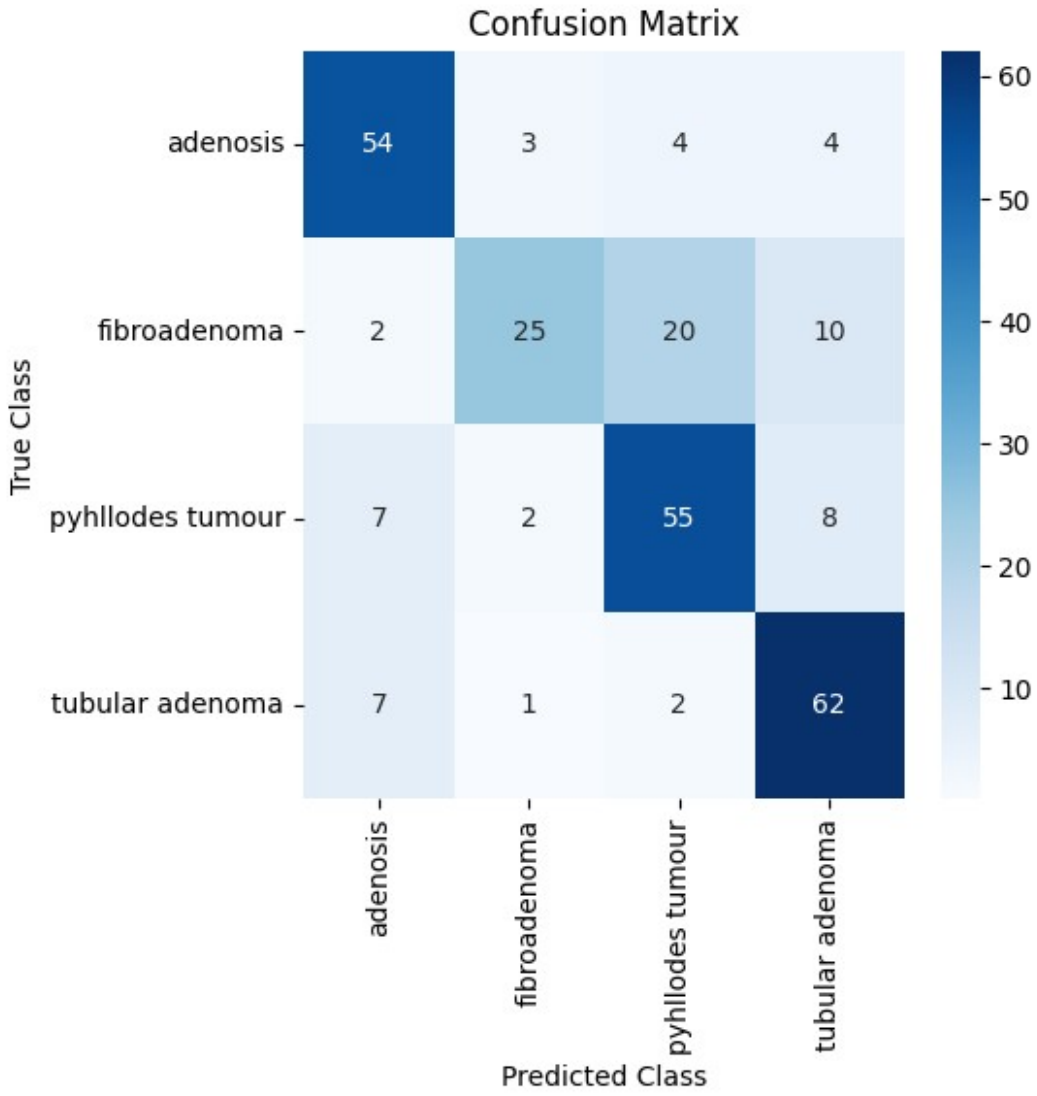



```
# Evaluate the model on the validation/test dataset
val_loss, val_accuracy = custom_model.evaluate(validation_ds)
print(f'Validation Loss: {val_loss}')
print(f'Validation Accuracy: {val_accuracy}')

# Predict the classes on the validation/test dataset
Y_pred = custom_model.predict(validation_ds)
y_pred = Y_pred.argmax(axis=1)  # Get the index of the max probability
(predicted class)
```

```
10/10 ━━━━━━━━━━━━━━━━━━━━ 3s 283ms/step - accuracy: 0.7311 - loss:
0.6295
Validation Loss: 0.6538627743721008
Validation Accuracy: 0.7368420958518982
10/10 ━━━━━━━━━━━━━━━━━━━━ 5s 373ms/step
```

```
#Let's now do a confusion matrix for each model to see how well they
perform

#1) OUR MODEL (CNN)
# Check if y_true needs to be converted
if y_true.ndim == 1:
    y_true_classes = y_true  # If it's already a vector of class
labels
else:
    y_true_classes = np.argmax(y_true, axis=1)  # If it's one-hot
encoded

# y_pred_classes is already computed correctly
```

```
y_pred_classes = np.argmax(Y_pred, axis=1)  # Predicted class labels

# Compute the confusion matrix
conf_matrix = confusion_matrix(y_true_classes, y_pred_classes)

# Plot the confusion matrix
class_names = ["adenosis", "fibroadenoma", "pyhllodes tumour","tubular
adenoma"]

plt.figure(figsize=(5, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Class')
plt.ylabel('True Class')
plt.show()
```



```python
from keras.utils import to_categorical
from sklearn.metrics import roc_auc_score

# Step 1: Get predicted probabilities from the model
y_pred_prob = custom_model.predict(X_test)  # Predicted probabilities
for each class

# Step 2: Convert y_test to one-hot encoding (if not already done)
y_test_one_hot = to_categorical(y_test, num_classes=4)

# Step 3: Calculate the ROC AUC score for multi-class classification
roc_auc = roc_auc_score(y_test_one_hot, y_pred_prob,
multi_class='ovr')
print(f"ROC AUC: {roc_auc}")
```

```
9/9 ━━━━━━━━━━━━━━━━━━━━ 4s 417ms/step
ROC AUC: 0.9166286142513758
```

```python
# Generate the Classification Report
#1) For CNN
report = classification_report(y_true_classes, y_pred_classes,
target_names=class_names)
print('Classification Report: CNN')
print(report)
```

```
Classification Report: CNN
                   precision    recall  f1-score   support

         adenosis       0.77      0.83      0.80        65
    fibroadenoma        0.81      0.44      0.57        57
pyhllodes tumour        0.68      0.76      0.72        72
 tubular adenoma        0.74      0.86      0.79        72

         accuracy                           0.74       266
        macro avg       0.75      0.72      0.72       266
    weighted avg        0.74      0.74      0.73       266
```

```python
# Load the ResNet50 model with pre-trained ImageNet weights, excluding
the top layer
ResNet_base_model = ResNet50(weights='imagenet', include_top=False,
input_shape=(224, 224, 3))

# Freeze the layers of the base model
for layer in ResNet_base_model.layers:
    layer.trainable = False

# Add custom layers on top of the base model
x = ResNet_base_model.output

#Global Average Pooling Layer
```

```
x = GlobalAveragePooling2D()(x)

#Fully Connected Layer
x = Dense(1024, activation='relu')(x)

#Output Layer
predictions = Dense(4, activation='softmax')(x)

# Create the final model
ResNetmodel = Model(inputs=ResNet_base_model.input,
outputs=predictions)

#Summary of the model
ResNetmodel.summary()
```

```
Model: "functional_3"
```

Layer (type)	Output Shape	Param #	Connected to
input_layer_2 (InputLayer)	(None, 224, 224, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 230, 230, 3)	0	input_layer_2[0][0]…
conv1_conv (Conv2D)	(None, 112, 112, 64)	9,472	conv1_pad[0][0]
conv1_bn (BatchNormalizatio…	(None, 112, 112, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 112, 112, 64)	0	conv1_bn[0][0]

```
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ pool1_pad            │ (None, 114, 114,   │           0 │ conv1_relu[0]
[0]  │
│ (ZeroPadding2D)      │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ pool1_pool           │ (None, 56, 56,     │           0 │ pool1_pad[0]
[0]   │
│ (MaxPooling2D)       │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_1_conv  │ (None, 56, 56,     │       4,160 │ pool1_pool[0]
[0]  │
│ (Conv2D)             │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_1_bn    │ (None, 56, 56,     │         256 │
conv2_block1_1_c… │
│ (BatchNormalizatio… │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_1_relu  │ (None, 56, 56,     │           0 │
conv2_block1_1_b… │
│ (Activation)         │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_2_conv  │ (None, 56, 56,     │      36,928 │
conv2_block1_1_r… │
│ (Conv2D)             │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_2_bn    │ (None, 56, 56,     │         256 │
conv2_block1_2_c… │
│ (BatchNormalizatio… │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_2_relu  │ (None, 56, 56,     │           0 │
conv2_block1_2_b… │
│ (Activation)         │ 64)                  │             │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_0_conv  │ (None, 56, 56,       │     16,640 │ pool1_pool[0]
[0]   │
│ (Conv2D)             │ 256)                 │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_3_conv  │ (None, 56, 56,       │     16,640 │
conv2_block1_2_r… │
│ (Conv2D)             │ 256)                 │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_0_bn    │ (None, 56, 56,       │      1,024 │
conv2_block1_0_c… │
│ (BatchNormalizatio… │ 256)                 │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_3_bn    │ (None, 56, 56,       │      1,024 │
conv2_block1_3_c… │
│ (BatchNormalizatio… │ 256)                 │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_add     │ (None, 56, 56,       │          0 │
conv2_block1_0_b… │
│ (Add)                │ 256)                 │            │
conv2_block1_3_b… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block1_out     │ (None, 56, 56,       │          0 │
conv2_block1_add… │
│ (Activation)         │ 256)                 │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block2_1_conv  │ (None, 56, 56,       │     16,448 │
conv2_block1_out… │
│ (Conv2D)             │ 64)                  │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block2_1_bn    │ (None, 56, 56,       │        256 │
conv2_block2_1_c… │
│ (BatchNormalizatio… │ 64)                  │            │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
```

```
──────┤
│ conv2_block2_1_relu │ (None, 56, 56,     │           0 │
conv2_block2_1_b… │
│ (Activation)        │ 64)                │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_2_conv │ (None, 56, 56,     │      36,928 │
conv2_block2_1_r… │
│ (Conv2D)            │ 64)                │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_2_bn   │ (None, 56, 56,     │         256 │
conv2_block2_2_c… │
│ (BatchNormalizatio… │ 64)                │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_2_relu │ (None, 56, 56,     │           0 │
conv2_block2_2_b… │
│ (Activation)        │ 64)                │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_3_conv │ (None, 56, 56,     │      16,640 │
conv2_block2_2_r… │
│ (Conv2D)            │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_3_bn   │ (None, 56, 56,     │       1,024 │
conv2_block2_3_c… │
│ (BatchNormalizatio… │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_add    │ (None, 56, 56,     │           0 │
conv2_block1_out… │
│ (Add)               │ 256)               │             │
conv2_block2_3_b… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_out    │ (None, 56, 56,     │           0 │
conv2_block2_add… │
│ (Activation)        │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| conv2_block3_1_conv | (None, 56, 56,     |     16,448 |
conv2_block2_out… |
| (Conv2D)            | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_1_bn   | (None, 56, 56,     |        256 |
conv2_block3_1_c… |
| (BatchNormalizatio… | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_1_relu | (None, 56, 56,     |          0 |
conv2_block3_1_b… |
| (Activation)        | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_2_conv | (None, 56, 56,     |     36,928 |
conv2_block3_1_r… |
| (Conv2D)            | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_2_bn   | (None, 56, 56,     |        256 |
conv2_block3_2_c… |
| (BatchNormalizatio… | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_2_relu | (None, 56, 56,     |          0 |
conv2_block3_2_b… |
| (Activation)        | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_3_conv | (None, 56, 56,     |     16,640 |
conv2_block3_2_r… |
| (Conv2D)            | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_3_bn   | (None, 56, 56,     |      1,024 |
conv2_block3_3_c… |
| (BatchNormalizatio… | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| conv2_block3_add    | (None, 56, 56,     |          0 |
```

```
conv2_block2_out… │
│ (Add)               │ 256)               │             │
conv2_block3_3_b… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block3_out    │ (None, 56, 56,     │           0 │
conv2_block3_add… │
│ (Activation)        │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_1_conv │ (None, 28, 28,     │      32,896 │
conv2_block3_out… │
│ (Conv2D)            │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_1_bn   │ (None, 28, 28,     │         512 │
conv3_block1_1_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_1_relu │ (None, 28, 28,     │           0 │
conv3_block1_1_b… │
│ (Activation)        │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_2_conv │ (None, 28, 28,     │     147,584 │
conv3_block1_1_r… │
│ (Conv2D)            │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_2_bn   │ (None, 28, 28,     │         512 │
conv3_block1_2_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_2_relu │ (None, 28, 28,     │           0 │
conv3_block1_2_b… │
│ (Activation)        │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block1_0_conv │ (None, 28, 28,     │     131,584 │
conv2_block3_out… │
```

```
| (Conv2D)              | 512)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block1_3_conv | (None, 28, 28,     |     66,048 |
conv3_block1_2_r… |
| (Conv2D)              | 512)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block1_0_bn   | (None, 28, 28,     |      2,048 |
conv3_block1_0_c… |
| (BatchNormalizatio… | 512)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block1_3_bn   | (None, 28, 28,     |      2,048 |
conv3_block1_3_c… |
| (BatchNormalizatio… | 512)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block1_add    | (None, 28, 28,     |           0 |
conv3_block1_0_b… |
| (Add)                 | 512)                 |             |
conv3_block1_3_b… |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block1_out    | (None, 28, 28,     |           0 |
conv3_block1_add… |
| (Activation)          | 512)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block2_1_conv | (None, 28, 28,     |     65,664 |
conv3_block1_out… |
| (Conv2D)              | 128)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block2_1_bn   | (None, 28, 28,     |         512 |
conv3_block2_1_c… |
| (BatchNormalizatio… | 128)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv3_block2_1_relu | (None, 28, 28,     |           0 |
conv3_block2_1_b… |
| (Activation)          | 128)                 |             |
```

```
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_2_conv │ (None, 28, 28,     │     147,584 │
conv3_block2_1_r… │
│ (Conv2D)            │ 128)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_2_bn   │ (None, 28, 28,     │         512 │
conv3_block2_2_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_2_relu │ (None, 28, 28,     │           0 │
conv3_block2_2_b… │
│ (Activation)        │ 128)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_3_conv │ (None, 28, 28,     │      66,048 │
conv3_block2_2_r… │
│ (Conv2D)            │ 512)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_3_bn   │ (None, 28, 28,     │       2,048 │
conv3_block2_3_c… │
│ (BatchNormalizatio… │ 512)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_add    │ (None, 28, 28,     │           0 │
conv3_block1_out… │
│ (Add)               │ 512)               │             │
conv3_block2_3_b… │
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block2_out    │ (None, 28, 28,     │           0 │
conv3_block2_add… │
│ (Activation)        │ 512)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼─────────────
──────┤
│ conv3_block3_1_conv │ (None, 28, 28,     │      65,664 │
conv3_block2_out… │
│ (Conv2D)            │ 128)               │             │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_1_bn    │ (None, 28, 28,       │          512 │
conv3_block3_1_c… │
│ (BatchNormalizatio… │ 128)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_1_relu  │ (None, 28, 28,       │            0 │
conv3_block3_1_b… │
│ (Activation)         │ 128)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_2_conv  │ (None, 28, 28,       │      147,584 │
conv3_block3_1_r… │
│ (Conv2D)             │ 128)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_2_bn    │ (None, 28, 28,       │          512 │
conv3_block3_2_c… │
│ (BatchNormalizatio… │ 128)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_2_relu  │ (None, 28, 28,       │            0 │
conv3_block3_2_b… │
│ (Activation)         │ 128)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_3_conv  │ (None, 28, 28,       │       66,048 │
conv3_block3_2_r… │
│ (Conv2D)             │ 512)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_3_bn    │ (None, 28, 28,       │        2,048 │
conv3_block3_3_c… │
│ (BatchNormalizatio… │ 512)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block3_add     │ (None, 28, 28,       │            0 │
conv3_block2_out… │
│ (Add)                │ 512)                 │              │
conv3_block3_3_b… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
```

```
──────┤
│ conv3_block3_out    │ (None, 28, 28,     │           0 │
conv3_block3_add… │
│ (Activation)        │ 512)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_1_conv │ (None, 28, 28,     │      65,664 │
conv3_block3_out… │
│ (Conv2D)            │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_1_bn   │ (None, 28, 28,     │         512 │
conv3_block4_1_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_1_relu │ (None, 28, 28,     │           0 │
conv3_block4_1_b… │
│ (Activation)        │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_2_conv │ (None, 28, 28,     │     147,584 │
conv3_block4_1_r… │
│ (Conv2D)            │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_2_bn   │ (None, 28, 28,     │         512 │
conv3_block4_2_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_2_relu │ (None, 28, 28,     │           0 │
conv3_block4_2_b… │
│ (Activation)        │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv3_block4_3_conv │ (None, 28, 28,     │      66,048 │
conv3_block4_2_r… │
│ (Conv2D)            │ 512)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| conv3_block4_3_bn   | (None, 28, 28,     |       2,048 |
conv3_block4_3_c… |
| (BatchNormalizatio… | 512)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_add    | (None, 28, 28,     |           0 |
conv3_block3_out… |
| (Add)               | 512)               |             |
conv3_block4_3_b… |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_out    | (None, 28, 28,     |           0 |
conv3_block4_add… |
| (Activation)        | 512)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block1_1_conv | (None, 14, 14,     |     131,328 |
conv3_block4_out… |
| (Conv2D)            | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block1_1_bn   | (None, 14, 14,     |       1,024 |
conv4_block1_1_c… |
| (BatchNormalizatio… | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block1_1_relu | (None, 14, 14,     |           0 |
conv4_block1_1_b… |
| (Activation)        | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block1_2_conv | (None, 14, 14,     |     590,080 |
conv4_block1_1_r… |
| (Conv2D)            | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block1_2_bn   | (None, 14, 14,     |       1,024 |
conv4_block1_2_c… |
| (BatchNormalizatio… | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block1_2_relu | (None, 14, 14,     |           0 |
conv4_block1_2_b… |
```

```
| (Activation)         | 256)                |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block1_0_conv | (None, 14, 14,     |    525,312 |
conv3_block4_out… |
| (Conv2D)             | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block1_3_conv | (None, 14, 14,     |    263,168 |
conv4_block1_2_r… |
| (Conv2D)             | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block1_0_bn    | (None, 14, 14,     |      4,096 |
conv4_block1_0_c… |
| (BatchNormalizatio… | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block1_3_bn    | (None, 14, 14,     |      4,096 |
conv4_block1_3_c… |
| (BatchNormalizatio… | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block1_add     | (None, 14, 14,     |          0 |
conv4_block1_0_b… |
| (Add)                | 1024)               |             |
conv4_block1_3_b… |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block1_out     | (None, 14, 14,     |          0 |
conv4_block1_add… |
| (Activation)         | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block2_1_conv | (None, 14, 14,     |    262,400 |
conv4_block1_out… |
| (Conv2D)             | 256)                |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────
──────┤
| conv4_block2_1_bn    | (None, 14, 14,     |      1,024 |
conv4_block2_1_c… |
| (BatchNormalizatio… | 256)                |             |
```

```
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_1_relu │ (None, 14, 14,   │            0 │
conv4_block2_1_b… │
│ (Activation)        │ 256)             │              │
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_2_conv │ (None, 14, 14,   │      590,080 │
conv4_block2_1_r… │
│ (Conv2D)            │ 256)             │              │
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_2_bn   │ (None, 14, 14,   │        1,024 │
conv4_block2_2_c… │
│ (BatchNormalizatio… │ 256)             │              │
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_2_relu │ (None, 14, 14,   │            0 │
conv4_block2_2_b… │
│ (Activation)        │ 256)             │              │
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_3_conv │ (None, 14, 14,   │      263,168 │
conv4_block2_2_r… │
│ (Conv2D)            │ 1024)            │              │
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_3_bn   │ (None, 14, 14,   │        4,096 │
conv4_block2_3_c… │
│ (BatchNormalizatio… │ 1024)            │              │
│
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_add    │ (None, 14, 14,   │            0 │
conv4_block1_out… │
│ (Add)               │ 1024)            │              │
conv4_block2_3_b… │
├─────────────────────┼──────────────────┼──────────────┼──────────────
─────┤
│ conv4_block2_out    │ (None, 14, 14,   │            0 │
conv4_block2_add… │
│ (Activation)        │ 1024)            │              │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_1_conv  │ (None, 14, 14,       │      262,400 │
conv4_block2_out… │
│ (Conv2D)             │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_1_bn    │ (None, 14, 14,       │        1,024 │
conv4_block3_1_c… │
│ (BatchNormalizatio… │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_1_relu  │ (None, 14, 14,       │            0 │
conv4_block3_1_b… │
│ (Activation)         │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_2_conv  │ (None, 14, 14,       │      590,080 │
conv4_block3_1_r… │
│ (Conv2D)             │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_2_bn    │ (None, 14, 14,       │        1,024 │
conv4_block3_2_c… │
│ (BatchNormalizatio… │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_2_relu  │ (None, 14, 14,       │            0 │
conv4_block3_2_b… │
│ (Activation)         │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_3_conv  │ (None, 14, 14,       │      263,168 │
conv4_block3_2_r… │
│ (Conv2D)             │ 1024)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
──────┤
│ conv4_block3_3_bn    │ (None, 14, 14,       │        4,096 │
conv4_block3_3_c… │
│ (BatchNormalizatio… │ 1024)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────
```

```
──────┤
| conv4_block3_add    | (None, 14, 14,    |           0 |
conv4_block2_out… |
| (Add)               | 1024)             |             |
conv4_block3_3_b… |
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block3_out    | (None, 14, 14,    |           0 |
conv4_block3_add… |
| (Activation)        | 1024)             |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block4_1_conv | (None, 14, 14,    |     262,400 |
conv4_block3_out… |
| (Conv2D)            | 256)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block4_1_bn   | (None, 14, 14,    |       1,024 |
conv4_block4_1_c… |
| (BatchNormalizatio… | 256)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block4_1_relu | (None, 14, 14,    |           0 |
conv4_block4_1_b… |
| (Activation)        | 256)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block4_2_conv | (None, 14, 14,    |     590,080 |
conv4_block4_1_r… |
| (Conv2D)            | 256)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block4_2_bn   | (None, 14, 14,    |       1,024 |
conv4_block4_2_c… |
| (BatchNormalizatio… | 256)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv4_block4_2_relu | (None, 14, 14,    |           0 |
conv4_block4_2_b… |
| (Activation)        | 256)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
```

```
| conv4_block4_3_conv | (None, 14, 14,     |    263,168 |
conv4_block4_2_r… |
| (Conv2D)            | 1024)              |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block4_3_bn   | (None, 14, 14,     |      4,096 |
conv4_block4_3_c… |
| (BatchNormalizatio… | 1024)              |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block4_add    | (None, 14, 14,     |          0 |
conv4_block3_out… |
| (Add)               | 1024)              |            |
conv4_block4_3_b… |
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block4_out    | (None, 14, 14,     |          0 |
conv4_block4_add… |
| (Activation)        | 1024)              |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block5_1_conv | (None, 14, 14,     |    262,400 |
conv4_block4_out… |
| (Conv2D)            | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block5_1_bn   | (None, 14, 14,     |      1,024 |
conv4_block5_1_c… |
| (BatchNormalizatio… | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block5_1_relu | (None, 14, 14,     |          0 |
conv4_block5_1_b… |
| (Activation)        | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block5_2_conv | (None, 14, 14,     |    590,080 |
conv4_block5_1_r… |
| (Conv2D)            | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv4_block5_2_bn   | (None, 14, 14,     |      1,024 |
```

```
conv4_block5_2_c… │
│ (BatchNormalizatio… │ 256)               │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block5_2_relu │ (None, 14, 14,     │          0 │
conv4_block5_2_b… │
│ (Activation)        │ 256)               │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block5_3_conv │ (None, 14, 14,     │    263,168 │
conv4_block5_2_r… │
│ (Conv2D)            │ 1024)              │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block5_3_bn   │ (None, 14, 14,     │      4,096 │
conv4_block5_3_c… │
│ (BatchNormalizatio… │ 1024)              │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block5_add    │ (None, 14, 14,     │          0 │
conv4_block4_out… │
│ (Add)               │ 1024)              │            │
conv4_block5_3_b… │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block5_out    │ (None, 14, 14,     │          0 │
conv4_block5_add… │
│ (Activation)        │ 1024)              │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block6_1_conv │ (None, 14, 14,     │    262,400 │
conv4_block5_out… │
│ (Conv2D)            │ 256)               │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block6_1_bn   │ (None, 14, 14,     │      1,024 │
conv4_block6_1_c… │
│ (BatchNormalizatio… │ 256)               │            │
├──────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ conv4_block6_1_relu │ (None, 14, 14,     │          0 │
conv4_block6_1_b… │
```

```
| (Activation)         | 256)                |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_2_conv | (None, 14, 14,     |    590,080 |
conv4_block6_1_r… |
| (Conv2D)             | 256)                |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_2_bn    | (None, 14, 14,     |      1,024 |
conv4_block6_2_c… |
| (BatchNormalizatio… | 256)                |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_2_relu | (None, 14, 14,     |          0 |
conv4_block6_2_b… |
| (Activation)         | 256)                |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_3_conv | (None, 14, 14,     |    263,168 |
conv4_block6_2_r… |
| (Conv2D)             | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_3_bn    | (None, 14, 14,     |      4,096 |
conv4_block6_3_c… |
| (BatchNormalizatio… | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_add     | (None, 14, 14,     |          0 |
conv4_block5_out… |
| (Add)                | 1024)               |             |
conv4_block6_3_b… |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv4_block6_out     | (None, 14, 14,     |          0 |
conv4_block6_add… |
| (Activation)         | 1024)               |             |
|                      |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| conv5_block1_1_conv | (None, 7, 7, 512) |    524,800 |
conv4_block6_out… |
| (Conv2D)             |                     |             |
```

```
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_1_bn   │ (None, 7, 7, 512)  │       2,048 │
conv5_block1_1_c… │
│ (BatchNormalizatio… │                    │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_1_relu │ (None, 7, 7, 512)  │           0 │
conv5_block1_1_b… │
│ (Activation)        │                    │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_2_conv │ (None, 7, 7, 512)  │   2,359,808 │
conv5_block1_1_r… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_2_bn   │ (None, 7, 7, 512)  │       2,048 │
conv5_block1_2_c… │
│ (BatchNormalizatio… │                    │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_2_relu │ (None, 7, 7, 512)  │           0 │
conv5_block1_2_b… │
│ (Activation)        │                    │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_0_conv │ (None, 7, 7,       │   2,099,200 │
conv4_block6_out… │
│ (Conv2D)            │ 2048)              │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_3_conv │ (None, 7, 7,       │   1,050,624 │
conv5_block1_2_r… │
│ (Conv2D)            │ 2048)              │             │
│
├─────────────────────┼──────────────────────┼──────────────┼─────────────
──────┤
│ conv5_block1_0_bn   │ (None, 7, 7,       │       8,192 │
conv5_block1_0_c… │
│ (BatchNormalizatio… │ 2048)              │             │
│
```

```
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block1_3_bn   │ (None, 7, 7,        │       8,192 │
conv5_block1_3_c… │
│ (BatchNormalizatio… │ 2048)               │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block1_add    │ (None, 7, 7,        │           0 │
conv5_block1_0_b… │
│ (Add)               │ 2048)               │             │
conv5_block1_3_b… │
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block1_out    │ (None, 7, 7,        │           0 │
conv5_block1_add… │
│ (Activation)        │ 2048)               │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_1_conv │ (None, 7, 7, 512)   │   1,049,088 │
conv5_block1_out… │
│ (Conv2D)            │                     │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_1_bn   │ (None, 7, 7, 512)   │       2,048 │
conv5_block2_1_c… │
│ (BatchNormalizatio… │                     │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_1_relu │ (None, 7, 7, 512)   │           0 │
conv5_block2_1_b… │
│ (Activation)        │                     │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_2_conv │ (None, 7, 7, 512)   │   2,359,808 │
conv5_block2_1_r… │
│ (Conv2D)            │                     │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_2_bn   │ (None, 7, 7, 512)   │       2,048 │
conv5_block2_2_c… │
│ (BatchNormalizatio… │                     │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
```

```
──────┤
│ conv5_block2_2_relu │ (None, 7, 7, 512) │          0 │
conv5_block2_2_b… │
│ (Activation)        │                   │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block2_3_conv │ (None, 7, 7,      │  1,050,624 │
conv5_block2_2_r… │
│ (Conv2D)            │ 2048)             │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block2_3_bn   │ (None, 7, 7,      │      8,192 │
conv5_block2_3_c… │
│ (BatchNormalizatio… │ 2048)             │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block2_add    │ (None, 7, 7,      │          0 │
conv5_block1_out… │
│ (Add)               │ 2048)             │            │
conv5_block2_3_b… │
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block2_out    │ (None, 7, 7,      │          0 │
conv5_block2_add… │
│ (Activation)        │ 2048)             │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block3_1_conv │ (None, 7, 7, 512) │  1,049,088 │
conv5_block2_out… │
│ (Conv2D)            │                   │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block3_1_bn   │ (None, 7, 7, 512) │      2,048 │
conv5_block3_1_c… │
│ (BatchNormalizatio… │                   │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ conv5_block3_1_relu │ (None, 7, 7, 512) │          0 │
conv5_block3_1_b… │
│ (Activation)        │                   │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
```

```
| conv5_block3_2_conv | (None, 7, 7, 512) |  2,359,808 |
conv5_block3_1_r… |
| (Conv2D)            |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block3_2_bn   | (None, 7, 7, 512) |      2,048 |
conv5_block3_2_c… |
| (BatchNormalizatio… |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block3_2_relu | (None, 7, 7, 512) |          0 |
conv5_block3_2_b… |
| (Activation)        |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block3_3_conv | (None, 7, 7,      |  1,050,624 |
conv5_block3_2_r… |
| (Conv2D)            | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block3_3_bn   | (None, 7, 7,      |      8,192 |
conv5_block3_3_c… |
| (BatchNormalizatio… | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block3_add    | (None, 7, 7,      |          0 |
conv5_block2_out… |
| (Add)               | 2048)             |            |
conv5_block3_3_b… |
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block3_out    | (None, 7, 7,      |          0 |
conv5_block3_add… |
| (Activation)        | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| global_average_poo… | (None, 2048)      |          0 |
conv5_block3_out… |
| (GlobalAveragePool… |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| dense_3 (Dense)     | (None, 1024)      |  2,098,176 |
global_average_p… |
```

```
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
─────┤
│ dense_4 (Dense)      │ (None, 4)            │       4,100 │ dense_3[0][0]
│
└──────────────────────┴──────────────────────┴─────────────┴──────────────────
─────┘

 Total params: 25,689,988 (98.00 MB)

 Trainable params: 2,102,276 (8.02 MB)

 Non-trainable params: 23,587,712 (89.98 MB)
```

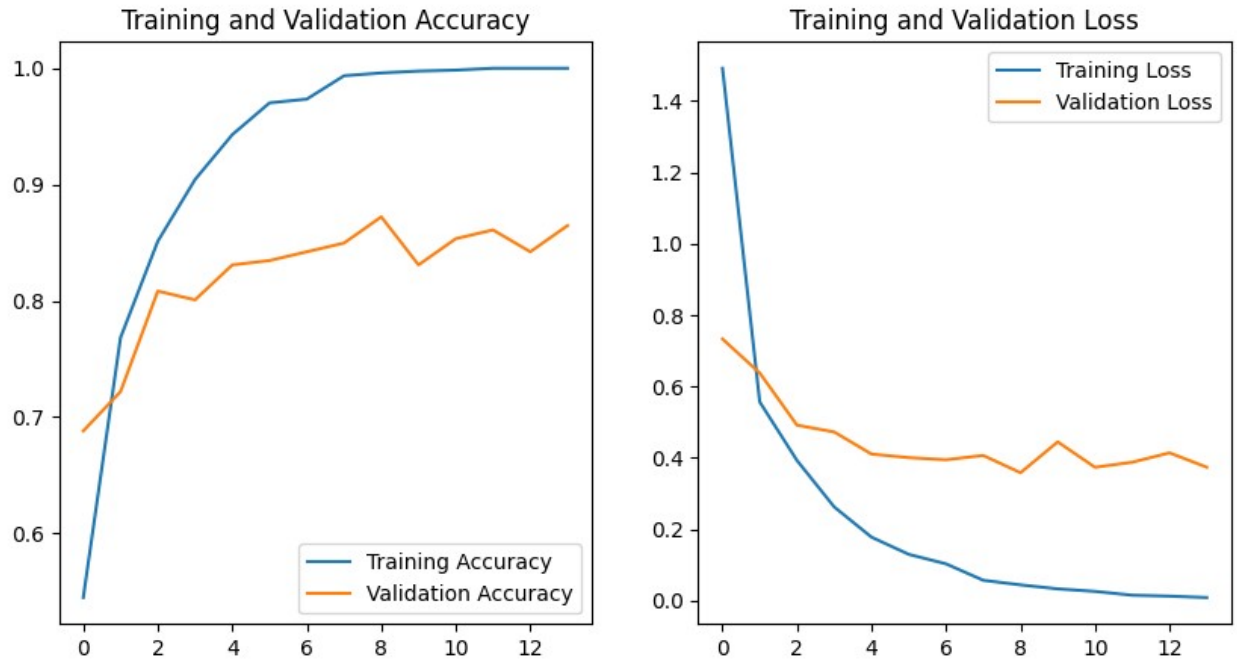
```
# Compile the model
ResNetmodel.compile(optimizer=Adam(learning_rate=0.001),
              loss='sparse_categorical_crossentropy',  # Suitable for
multi-class classification
              metrics=["accuracy"])

#Fit the model
epochs=20
history = ResNetmodel.fit(train_ds, validation_data=validation_ds,
epochs=epochs, callbacks = [early_stopping, reduce_lr])
```

```
Epoch 1/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 171s 4s/step - accuracy: 0.4238 - loss:
2.2713 - val_accuracy: 0.6880 - val_loss: 0.7333 - learning_rate:
0.0010
Epoch 2/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 148s 3s/step - accuracy: 0.7611 - loss:
0.5886 - val_accuracy: 0.7218 - val_loss: 0.6381 - learning_rate:
0.0010
Epoch 3/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 145s 3s/step - accuracy: 0.8587 - loss:
0.3901 - val_accuracy: 0.8083 - val_loss: 0.4919 - learning_rate:
0.0010
Epoch 4/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 118s 3s/step - accuracy: 0.8913 - loss:
0.2815 - val_accuracy: 0.8008 - val_loss: 0.4728 - learning_rate:
0.0010
Epoch 5/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 104s 2s/step - accuracy: 0.9418 - loss:
0.1807 - val_accuracy: 0.8308 - val_loss: 0.4109 - learning_rate:
0.0010
Epoch 6/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 123s 3s/step - accuracy: 0.9745 - loss:
0.1252 - val_accuracy: 0.8346 - val_loss: 0.4010 - learning_rate:
0.0010
Epoch 7/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 124s 3s/step - accuracy: 0.9750 - loss:
0.0983 - val_accuracy: 0.8421 - val_loss: 0.3950 - learning_rate:
```

```
0.0010
Epoch 8/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 128s 3s/step - accuracy: 0.9950 - loss:
0.0595 - val_accuracy: 0.8496 - val_loss: 0.4070 - learning_rate:
0.0010
Epoch 9/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 141s 3s/step - accuracy: 0.9949 - loss:
0.0458 - val_accuracy: 0.8722 - val_loss: 0.3586 - learning_rate:
0.0010
Epoch 10/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 143s 3s/step - accuracy: 0.9990 - loss:
0.0308 - val_accuracy: 0.8308 - val_loss: 0.4451 - learning_rate:
0.0010
Epoch 11/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 193s 4s/step - accuracy: 0.9970 - loss:
0.0280 - val_accuracy: 0.8534 - val_loss: 0.3739 - learning_rate:
0.0010
Epoch 12/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 144s 3s/step - accuracy: 1.0000 - loss:
0.0157 - val_accuracy: 0.8609 - val_loss: 0.3881 - learning_rate:
0.0010
Epoch 13/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 1.0000 - loss:
0.0161
Epoch 13: ReduceLROnPlateau reducing learning rate to
0.0005000000237487257.
45/45 ━━━━━━━━━━━━━━━━━━━━ 142s 3s/step - accuracy: 1.0000 - loss:
0.0160 - val_accuracy: 0.8421 - val_loss: 0.4143 - learning_rate:
0.0010
Epoch 14/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 160s 4s/step - accuracy: 1.0000 - loss:
0.0088 - val_accuracy: 0.8647 - val_loss: 0.3741 - learning_rate:
5.0000e-04
Epoch 14: early stopping
Restoring model weights from the end of the best epoch: 9.
```

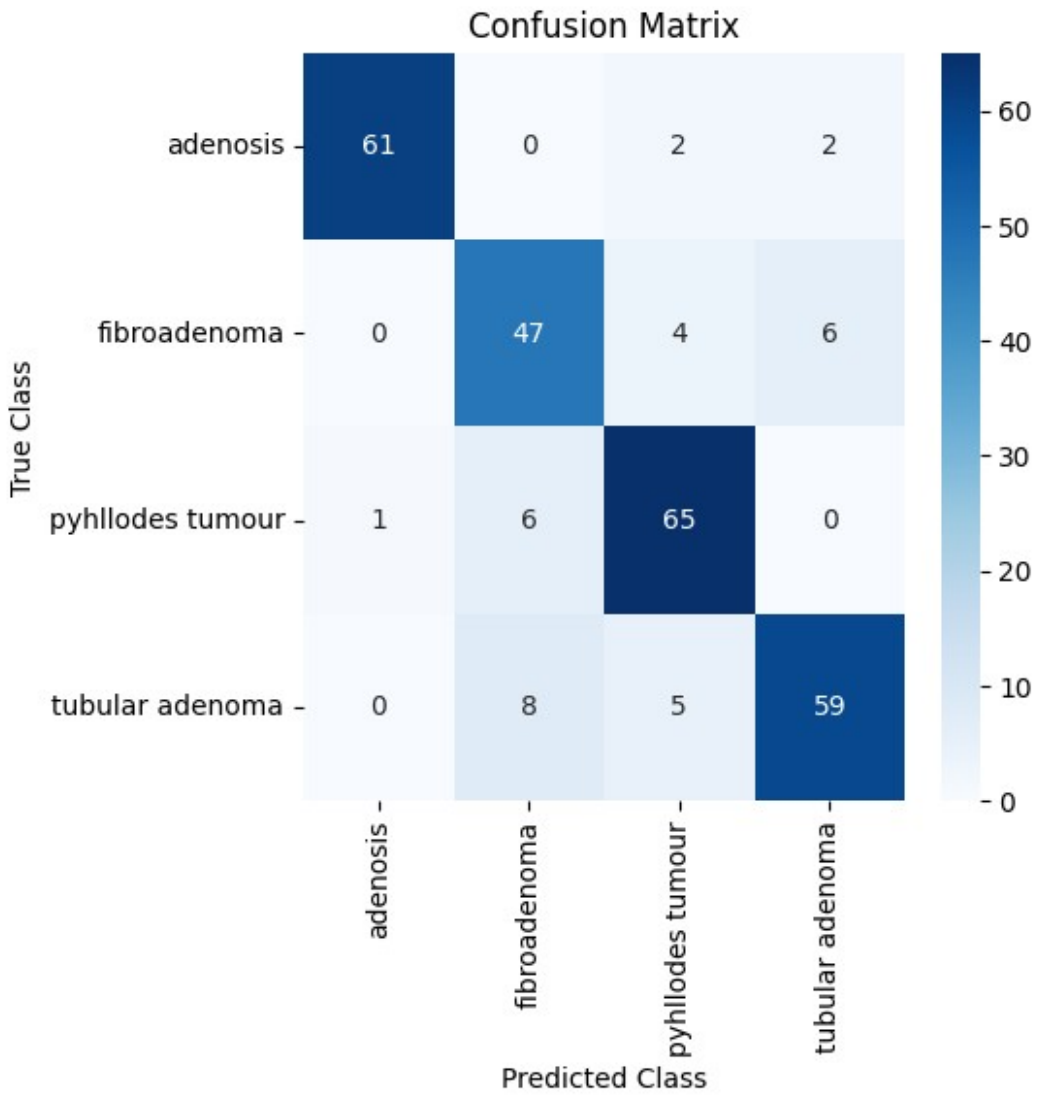
```
ResNet_acc = history.history['accuracy']
ResNet_val_acc = history.history['val_accuracy']

ResNet_loss = history.history['loss']
ResNet_val_loss = history.history['val_loss']

epochs_range = range(len(history.history['accuracy']))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, ResNet_acc, label='Training Accuracy')
plt.plot(epochs_range, ResNet_val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')
```

```
plt.subplot(1, 2, 2)
plt.plot(epochs_range, ResNet_loss, label='Training Loss')
plt.plot(epochs_range, ResNet_val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```




```
# Evaluate the model on the validation/test dataset
ResNet_val_loss, ResNet_val_accuracy =
ResNetmodel.evaluate(validation_ds)
print(f'ResNet Validation Loss: {ResNet_val_loss}')
print(f'ResNet Validation Accuracy: {ResNet_val_accuracy}')

# Predict the classes on the validation/test dataset
ResNet_Y_pred = ResNetmodel.predict(validation_ds)
ResNet_y_pred = ResNet_Y_pred.argmax(axis=1)  # Get the index of the
max probability (predicted class)

10/10 ━━━━━━━━━━━━━━━━━━━━ 33s 3s/step - accuracy: 0.8569 - loss:
0.3990
ResNet Validation Loss: 0.35862991213798523
ResNet Validation Accuracy: 0.8721804618835449
10/10 ━━━━━━━━━━━━━━━━━━━━ 41s 4s/step

#2) ResNet
#  Check if y_true needs to be converted
if y_true.ndim == 1:
    ResNet_y_true_classes = y_true  # If it's already a vector of
```

```
class labels
else:
    ResNet_y_true_classes = np.argmax(y_true, axis=1)  # If it's one-
hot encoded

# y_pred_classes is already computed correctly
ResNet_y_pred_classes = np.argmax(ResNet_Y_pred, axis=1)  # Predicted
class labels

# Compute the confusion matrix
conf_matrix = confusion_matrix(ResNet_y_true_classes,
ResNet_y_pred_classes)

# Plot the confusion matrix
# Plot the confusion matrix
class_names = ["adenosis", "fibroadenoma", "pyhllodes tumour","tubular
adenoma"]

plt.figure(figsize=(5, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Class')
plt.ylabel('True Class')
plt.show()
```




```
#predicted probabilities for each class
y_pred_prob = ResNetmodel.predict(X_test)

# Convert y_test to one-hot encoded
y_test_one_hot = to_categorical(y_test, num_classes=4)

# Calculate the ROC AUC score
roc_auc = roc_auc_score(y_test_one_hot, y_pred_prob,
multi_class='ovr')
print(f"ROC AUC: {roc_auc}")
```

```
9/9 ━━━━━━━━━━━━━━━━━━━━ 44s 4s/step
ROC AUC: 0.9572757995160028
```

```
# Classification Report
#2) ResNet
```

```
report = classification_report(ResNet_y_true_classes,
ResNet_y_pred_classes, target_names=class_names)
print('Classification Report: ResNet')
print(report)
```

```
Classification Report: ResNet
                  precision    recall  f1-score   support

        adenosis       0.98      0.94      0.96        65
    fibroadenoma       0.77      0.82      0.80        57
pyhllodes tumour       0.86      0.90      0.88        72
 tubular adenoma       0.88      0.82      0.85        72

        accuracy                           0.87       266
       macro avg       0.87      0.87      0.87       266
    weighted avg       0.88      0.87      0.87       266
```

```
# Load the ResNet50 model with pre-trained ImageNet weights, excluding
the top layer
# Load the EfficientNetB0 model with pre-trained ImageNet weights,
excluding the top layer
EfficientNetB0_base_model = EfficientNetB0(weights='imagenet',
include_top=False, input_shape=(224, 224, 3))


# Freeze the layers of the base model
for layer in EfficientNetB0_base_model.layers:
    layer.trainable = False

# Add custom layers on top of the base model
x = EfficientNetB0_base_model.output

#Global Average Pooling Layer
x = GlobalAveragePooling2D()(x)

#Fully Connected Layer
x = Dense(1024, activation='relu')(x)

#Output Layer
predictions = Dense(4, activation='softmax')(x)

# Create the final model
EfficientNetB0model = Model(inputs=EfficientNetB0_base_model.input,
outputs=predictions)

#Summary of the model
EfficientNetB0model.summary()
```

```
Model: "functional_5"
```

```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                    ┃ Output Shape           ┃       Param # ┃ Connected to               ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ input_layer_3 (InputLayer)      │ (None, 224, 224, 3)    │             0 │ -                          │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ rescaling_2 (Rescaling)         │ (None, 224, 224, 3)    │             0 │ input_layer_3[0][0]…       │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ normalization (Normalization)   │ (None, 224, 224, 3)    │             7 │ rescaling_2[0][0]          │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ rescaling_3 (Rescaling)         │ (None, 224, 224, 3)    │             0 │ normalization[0][0]…       │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ stem_conv_pad (ZeroPadding2D)   │ (None, 225, 225, 3)    │             0 │ rescaling_3[0][0]          │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ stem_conv (Conv2D)              │ (None, 112, 112, 32)   │           864 │ stem_conv_pad[0][0]…       │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ stem_bn (BatchNormalizatio…     │ (None, 112, 112, 32)   │           128 │ stem_conv[0][0]            │
├─────────────────────────────────┼────────────────────────┼───────────────┼────────────────────────────┤
│ stem_activation                 │ (None, 112, 112,       │             0 │ stem_bn[0][0]              │
```

```
| (Activation)         | 32)                 |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_dwconv       | (None, 112, 112,    |         288 |
stem_activation[… |
| (DepthwiseConv2D)    | 32)                 |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_bn           | (None, 112, 112,    |         128 |
block1a_dwconv[0… |
| (BatchNormalizatio… | 32)                 |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_activation   | (None, 112, 112,    |           0 | block1a_bn[0]
[0]  |
| (Activation)         | 32)                 |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_se_squeeze   | (None, 32)          |           0 |
block1a_activati… |
| (GlobalAveragePool… |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_se_reshape   | (None, 1, 1, 32)    |           0 |
block1a_se_squee… |
| (Reshape)            |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_se_reduce    | (None, 1, 1, 8)     |         264 |
block1a_se_resha… |
| (Conv2D)             |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_se_expand    | (None, 1, 1, 32)    |         288 |
block1a_se_reduc… |
| (Conv2D)             |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block1a_se_excite    | (None, 112, 112,    |           0 |
block1a_activati… |
| (Multiply)           | 32)                 |             |
block1a_se_expan… |
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block1a_project_co… │ (None, 112, 112,  │          512 │ block1a_se_excit… │
│ (Conv2D)             │ 16)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block1a_project_bn   │ (None, 112, 112,  │           64 │ block1a_project_… │
│ (BatchNormalizatio… │ 16)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block2a_expand_conv  │ (None, 112, 112,  │        1,536 │ block1a_project_… │
│ (Conv2D)             │ 96)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block2a_expand_bn    │ (None, 112, 112,  │          384 │ block2a_expand_c… │
│ (BatchNormalizatio… │ 96)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block2a_expand_act… │ (None, 112, 112,  │            0 │ block2a_expand_b… │
│ (Activation)         │ 96)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block2a_dwconv_pad   │ (None, 113, 113,  │            0 │ block2a_expand_a… │
│ (ZeroPadding2D)      │ 96)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block2a_dwconv       │ (None, 56, 56,    │          864 │ block2a_dwconv_p… │
│ (DepthwiseConv2D)    │ 96)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
│ block2a_bn           │ (None, 56, 56,    │          384 │ block2a_dwconv[0… │
│ (BatchNormalizatio… │ 96)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────────────┤
```

```
| block2a_activation  | (None, 56, 56,     |           0 | block2a_bn[0]
[0]  |
| (Activation)        | 96)                |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_squeeze  | (None, 96)         |           0 |
block2a_activati… |
| (GlobalAveragePool… |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_reshape  | (None, 1, 1, 96)   |           0 |
block2a_se_squee… |
| (Reshape)           |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_reduce   | (None, 1, 1, 4)    |         388 |
block2a_se_resha… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_expand   | (None, 1, 1, 96)   |         480 |
block2a_se_reduc… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_excite   | (None, 56, 56,     |           0 |
block2a_activati… |
| (Multiply)          | 96)                |             |
block2a_se_expan… |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_project_co… | (None, 56, 56,     |       2,304 |
block2a_se_excit… |
| (Conv2D)            | 24)                |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2a_project_bn  | (None, 56, 56,     |          96 |
block2a_project_… |
| (BatchNormalizatio… | 24)                |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block2b_expand_conv | (None, 56, 56,     |       3,456 |
block2a_project_… |
```

```
| (Conv2D)              | 144)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_expand_bn     | (None, 56, 56,       |         576 |
block2b_expand_c… |
| (BatchNormalizatio… | 144)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_expand_act… | (None, 56, 56,       |           0 |
block2b_expand_b… |
| (Activation)          | 144)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_dwconv        | (None, 56, 56,       |       1,296 |
block2b_expand_a… |
| (DepthwiseConv2D)     | 144)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_bn            | (None, 56, 56,       |         576 |
block2b_dwconv[0… |
| (BatchNormalizatio… | 144)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_activation    | (None, 56, 56,       |           0 | block2b_bn[0]
[0]   |
| (Activation)          | 144)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_se_squeeze    | (None, 144)          |           0 |
block2b_activati… |
| (GlobalAveragePool… |                      |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_se_reshape    | (None, 1, 1, 144)    |           0 |
block2b_se_squee… |
| (Reshape)             |                      |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block2b_se_reduce     | (None, 1, 1, 6)      |         870 |
block2b_se_resha… |
| (Conv2D)              |                      |             |
|
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_se_expand    │ (None, 1, 1, 144)    │        1,008 │
block2b_se_reduc… │
│ (Conv2D)             │                      │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_se_excite    │ (None, 56, 56,       │            0 │
block2b_activati… │
│ (Multiply)           │ 144)                 │              │
block2b_se_expan… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_project_co… │ (None, 56, 56,       │        3,456 │
block2b_se_excit… │
│ (Conv2D)             │ 24)                  │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_project_bn   │ (None, 56, 56,       │           96 │
block2b_project_… │
│ (BatchNormalizatio… │ 24)                  │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_drop         │ (None, 56, 56,       │            0 │
block2b_project_… │
│ (Dropout)            │ 24)                  │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_add (Add)    │ (None, 56, 56,       │            0 │
block2b_drop[0][… │
│                      │ 24)                  │              │
block2a_project_… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3a_expand_conv  │ (None, 56, 56,       │        3,456 │
block2b_add[0][0] │
│ (Conv2D)             │ 144)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3a_expand_bn    │ (None, 56, 56,       │          576 │
block3a_expand_c… │
│ (BatchNormalizatio… │ 144)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
```

```
| block3a_expand_act… | (None, 56, 56,     |           0 |
block3a_expand_b… |
| (Activation)        | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_dwconv_pad  | (None, 59, 59,     |           0 |
block3a_expand_a… |
| (ZeroPadding2D)     | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_dwconv      | (None, 28, 28,     |       3,600 |
block3a_dwconv_p… |
| (DepthwiseConv2D)   | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_bn          | (None, 28, 28,     |         576 |
block3a_dwconv[0… |
| (BatchNormalizatio… | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_activation  | (None, 28, 28,     |           0 | block3a_bn[0]
[0]  |
| (Activation)        | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_se_squeeze  | (None, 144)        |           0 |
block3a_activati… |
| (GlobalAveragePool… |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_se_reshape  | (None, 1, 1, 144)  |           0 |
block3a_se_squee… |
| (Reshape)           |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_se_reduce   | (None, 1, 1, 6)    |         870 |
block3a_se_resha… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3a_se_expand   | (None, 1, 1, 144)  |       1,008 |
block3a_se_reduc… |
```

```
│ (Conv2D)              │                    │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3a_se_excite     │ (None, 28, 28,     │            0 │
block3a_activati… │
│ (Multiply)            │ 144)               │              │
block3a_se_expan… │
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3a_project_co… │ (None, 28, 28,     │        5,760 │
block3a_se_excit… │
│ (Conv2D)              │ 40)                │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3a_project_bn    │ (None, 28, 28,     │          160 │
block3a_project_… │
│ (BatchNormalizatio… │ 40)                │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3b_expand_conv │ (None, 28, 28,     │        9,600 │
block3a_project_… │
│ (Conv2D)              │ 240)               │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3b_expand_bn     │ (None, 28, 28,     │          960 │
block3b_expand_c… │
│ (BatchNormalizatio… │ 240)               │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3b_expand_act… │ (None, 28, 28,     │            0 │
block3b_expand_b… │
│ (Activation)          │ 240)               │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3b_dwconv        │ (None, 28, 28,     │        6,000 │
block3b_expand_a… │
│ (DepthwiseConv2D)     │ 240)               │              │
│
├───────────────────────┼────────────────────┼──────────────┼──────────────
───────┤
│ block3b_bn            │ (None, 28, 28,     │          960 │
block3b_dwconv[0… │
│ (BatchNormalizatio… │ 240)               │              │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_activation   │ (None, 28, 28,       │            0 │ block3b_bn[0]
[0]   │
│ (Activation)         │ 240)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_se_squeeze   │ (None, 240)          │            0 │
block3b_activati… │
│ (GlobalAveragePool… │                      │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_se_reshape   │ (None, 1, 1, 240)    │            0 │
block3b_se_squee… │
│ (Reshape)            │                      │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_se_reduce    │ (None, 1, 1, 10)     │        2,410 │
block3b_se_resha… │
│ (Conv2D)             │                      │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_se_expand    │ (None, 1, 1, 240)    │        2,640 │
block3b_se_reduc… │
│ (Conv2D)             │                      │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_se_excite    │ (None, 28, 28,       │            0 │
block3b_activati… │
│ (Multiply)           │ 240)                 │              │
block3b_se_expan… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_project_co… │ (None, 28, 28,       │        9,600 │
block3b_se_excit… │
│ (Conv2D)             │ 40)                  │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block3b_project_bn   │ (None, 28, 28,       │          160 │
block3b_project_… │
│ (BatchNormalizatio… │ 40)                  │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
```

```
| block3b_drop         | (None, 28, 28,     |           0 |
block3b_project_… |
| (Dropout)            | 40)                |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block3b_add (Add)    | (None, 28, 28,     |           0 |
block3b_drop[0][… |
|                      | 40)                |             |
block3a_project_… |
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_expand_conv | (None, 28, 28,     |       9,600 |
block3b_add[0][0] |
| (Conv2D)             | 240)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_expand_bn    | (None, 28, 28,     |         960 |
block4a_expand_c… |
| (BatchNormalizatio… | 240)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_expand_act… | (None, 28, 28,     |           0 |
block4a_expand_b… |
| (Activation)         | 240)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_dwconv_pad   | (None, 29, 29,     |           0 |
block4a_expand_a… |
| (ZeroPadding2D)      | 240)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_dwconv       | (None, 14, 14,     |       2,160 |
block4a_dwconv_p… |
| (DepthwiseConv2D)    | 240)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_bn           | (None, 14, 14,     |         960 |
block4a_dwconv[0… |
| (BatchNormalizatio… | 240)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4a_activation   | (None, 14, 14,     |           0 | block4a_bn[0]
[0]   |
```

```
| (Activation)         | 240)                | |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_se_squeeze   | (None, 240)         |           0 |
block4a_activati… |
| (GlobalAveragePool… |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_se_reshape   | (None, 1, 1, 240) |           0 |
block4a_se_squee… |
| (Reshape)            |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_se_reduce    | (None, 1, 1, 10)  |       2,410 |
block4a_se_resha… |
| (Conv2D)             |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_se_expand    | (None, 1, 1, 240) |       2,640 |
block4a_se_reduc… |
| (Conv2D)             |                     |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_se_excite    | (None, 14, 14,      |           0 |
block4a_activati… |
| (Multiply)           | 240)                |             |
block4a_se_expan… |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_project_co… | (None, 14, 14,      |      19,200 |
block4a_se_excit… |
| (Conv2D)             | 80)                 |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4a_project_bn   | (None, 14, 14,      |         320 |
block4a_project_… |
| (BatchNormalizatio… | 80)                 |             |
|──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| block4b_expand_conv | (None, 14, 14,      |      38,400 |
block4a_project_… |
| (Conv2D)             | 480)                |             |
|
```

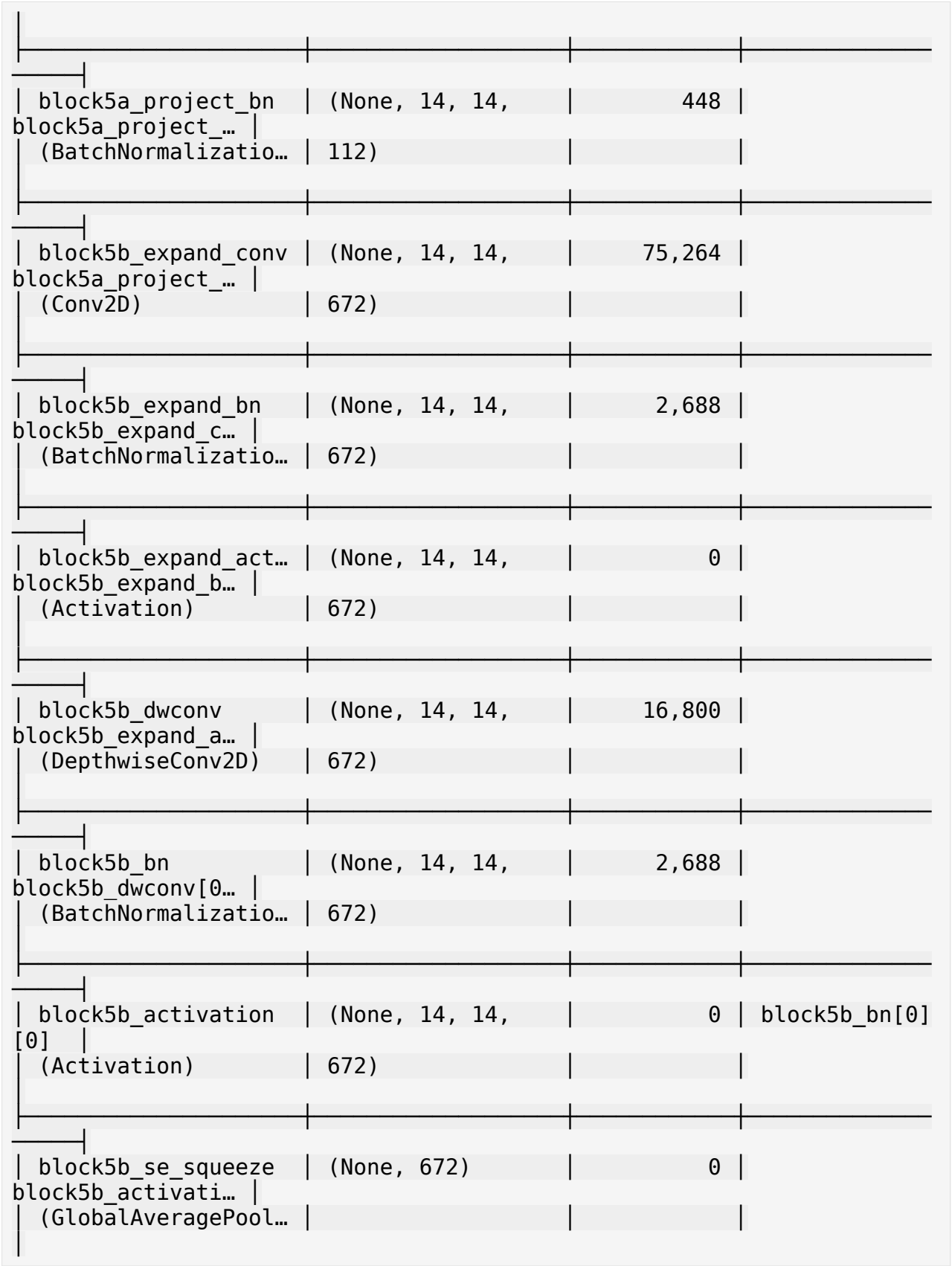
```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_expand_bn    │ (None, 14, 14,       │       1,920 │ block4b_expand_c… │
│ (BatchNormalizatio… │ 480)                 │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_expand_act… │ (None, 14, 14,       │           0 │ block4b_expand_b… │
│ (Activation)         │ 480)                 │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_dwconv       │ (None, 14, 14,       │       4,320 │ block4b_expand_a… │
│ (DepthwiseConv2D)    │ 480)                 │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_bn           │ (None, 14, 14,       │       1,920 │ block4b_dwconv[0… │
│ (BatchNormalizatio… │ 480)                 │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_activation   │ (None, 14, 14,       │           0 │ block4b_bn[0][0]  │
│ (Activation)         │ 480)                 │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_se_squeeze   │ (None, 480)          │           0 │ block4b_activati… │
│ (GlobalAveragePool… │                      │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_se_reshape   │ (None, 1, 1, 480)    │           0 │ block4b_se_squee… │
│ (Reshape)            │                      │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
│ block4b_se_reduce    │ (None, 1, 1, 20)     │       9,620 │ block4b_se_resha… │
│ (Conv2D)             │                      │             │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┼──────┤
```

```
──────┤
│ block4b_se_expand   │ (None, 1, 1, 480) │     10,080 │
block4b_se_reduc… │
│ (Conv2D)            │                   │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4b_se_excite   │ (None, 14, 14,    │          0 │
block4b_activati… │
│ (Multiply)          │ 480)              │            │
block4b_se_expan… │
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4b_project_co… │ (None, 14, 14,    │     38,400 │
block4b_se_excit… │
│ (Conv2D)            │ 80)               │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4b_project_bn  │ (None, 14, 14,    │        320 │
block4b_project_… │
│ (BatchNormalizatio… │ 80)               │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4b_drop        │ (None, 14, 14,    │          0 │
block4b_project_… │
│ (Dropout)           │ 80)               │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4b_add (Add)   │ (None, 14, 14,    │          0 │
block4b_drop[0][… │
│                     │ 80)               │            │
block4a_project_… │
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4c_expand_conv │ (None, 14, 14,    │     38,400 │
block4b_add[0][0] │
│ (Conv2D)            │ 480)              │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
│ block4c_expand_bn   │ (None, 14, 14,    │      1,920 │
block4c_expand_c… │
│ (BatchNormalizatio… │ 480)              │            │
│
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
```

```
| block4c_expand_act… | (None, 14, 14,     |           0 |
block4c_expand_b… |
| (Activation)        | 480)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_dwconv      | (None, 14, 14,     |       4,320 |
block4c_expand_a… |
| (DepthwiseConv2D)   | 480)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_bn          | (None, 14, 14,     |       1,920 |
block4c_dwconv[0… |
| (BatchNormalizatio… | 480)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_activation  | (None, 14, 14,     |           0 | block4c_bn[0]
[0]  |
| (Activation)        | 480)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_se_squeeze  | (None, 480)        |           0 |
block4c_activati… |
| (GlobalAveragePool… |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_se_reshape  | (None, 1, 1, 480)  |           0 |
block4c_se_squee… |
| (Reshape)           |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_se_reduce   | (None, 1, 1, 20)   |       9,620 |
block4c_se_resha… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_se_expand   | (None, 1, 1, 480)  |      10,080 |
block4c_se_reduc… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| block4c_se_excite   | (None, 14, 14,     |           0 |
```

```
block4c_activati… │
│ (Multiply)          │ 480)                │             │
block4c_se_expan… │
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_project_co… │ (None, 14, 14,      │     38,400 │
block4c_se_excit… │
│ (Conv2D)            │ 80)                 │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_project_bn  │ (None, 14, 14,      │        320 │
block4c_project_… │
│ (BatchNormalizatio… │ 80)                 │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_drop        │ (None, 14, 14,      │          0 │
block4c_project_… │
│ (Dropout)           │ 80)                 │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_add (Add)   │ (None, 14, 14,      │          0 │
block4c_drop[0][… │
│                     │ 80)                 │             │
block4b_add[0][0] │
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block5a_expand_conv │ (None, 14, 14,      │     38,400 │
block4c_add[0][0] │
│ (Conv2D)            │ 480)                │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block5a_expand_bn   │ (None, 14, 14,      │      1,920 │
block5a_expand_c… │
│ (BatchNormalizatio… │ 480)                │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block5a_expand_act… │ (None, 14, 14,      │          0 │
block5a_expand_b… │
│ (Activation)        │ 480)                │             │
│
├─────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block5a_dwconv      │ (None, 14, 14,      │     12,000 │
block5a_expand_a… │
```

(DepthwiseConv2D)	480)		
block5a_bn (BatchNormalizatio…	(None, 14, 14, 480)	1,920	block5a_dwconv[0…
block5a_activation (Activation)	(None, 14, 14, 480)	0	block5a_bn[0][0]
block5a_se_squeeze (GlobalAveragePool…	(None, 480)	0	block5a_activati…
block5a_se_reshape (Reshape)	(None, 1, 1, 480)	0	block5a_se_squee…
block5a_se_reduce (Conv2D)	(None, 1, 1, 20)	9,620	block5a_se_resha…
block5a_se_expand (Conv2D)	(None, 1, 1, 480)	10,080	block5a_se_reduc…
block5a_se_excite (Multiply)	(None, 14, 14, 480)	0	block5a_activati… block5a_se_expan…
block5a_project_co… (Conv2D)	(None, 14, 14, 112)	53,760	block5a_se_excit…



```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_se_reshape   │ (None, 1, 1, 672)    │            0 │ block5b_se_squee…    │
│ (Reshape)            │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_se_reduce    │ (None, 1, 1, 28)     │       18,844 │ block5b_se_resha…    │
│ (Conv2D)             │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_se_expand    │ (None, 1, 1, 672)    │       19,488 │ block5b_se_reduc…    │
│ (Conv2D)             │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_se_excite    │ (None, 14, 14,       │            0 │ block5b_activati…    │
│ (Multiply)           │ 672)                 │              │ block5b_se_expan…    │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_project_co…  │ (None, 14, 14,       │       75,264 │ block5b_se_excit…    │
│ (Conv2D)             │ 112)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_project_bn   │ (None, 14, 14,       │          448 │ block5b_project_…    │
│ (BatchNormalizatio…  │ 112)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_drop         │ (None, 14, 14,       │            0 │ block5b_project_…    │
│ (Dropout)            │ 112)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block5b_add (Add)    │ (None, 14, 14,       │            0 │ block5b_drop[0][…    │
│                      │ 112)                 │              │ block5a_project_…    │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
```

```
──────┤
| block5c_expand_conv | (None, 14, 14,     |     75,264 |
block5b_add[0][0] |
| (Conv2D)            | 672)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_expand_bn   | (None, 14, 14,     |      2,688 |
block5c_expand_c… |
| (BatchNormalizatio… | 672)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_expand_act… | (None, 14, 14,     |          0 |
block5c_expand_b… |
| (Activation)        | 672)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_dwconv      | (None, 14, 14,     |     16,800 |
block5c_expand_a… |
| (DepthwiseConv2D)   | 672)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_bn          | (None, 14, 14,     |      2,688 |
block5c_dwconv[0… |
| (BatchNormalizatio… | 672)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_activation  | (None, 14, 14,     |          0 | block5c_bn[0]
[0]  |
| (Activation)        | 672)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_se_squeeze  | (None, 672)        |          0 |
block5c_activati… |
| (GlobalAveragePool… |                    |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block5c_se_reshape  | (None, 1, 1, 672)  |          0 |
block5c_se_squee… |
| (Reshape)           |                    |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
```

```
| block5c_se_reduce   | (None, 1, 1, 28)  |     18,844 |
block5c_se_resha… |
| (Conv2D)            |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block5c_se_expand   | (None, 1, 1, 672) |     19,488 |
block5c_se_reduc… |
| (Conv2D)            |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block5c_se_excite   | (None, 14, 14,    |          0 |
block5c_activati… |
| (Multiply)          | 672)              |            |
block5c_se_expan… |
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block5c_project_co… | (None, 14, 14,    |     75,264 |
block5c_se_excit… |
| (Conv2D)            | 112)              |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block5c_project_bn  | (None, 14, 14,    |        448 |
block5c_project_… |
| (BatchNormalizatio… | 112)              |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block5c_drop        | (None, 14, 14,    |          0 |
block5c_project_… |
| (Dropout)           | 112)              |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block5c_add (Add)   | (None, 14, 14,    |          0 |
block5c_drop[0][… |
|                     | 112)              |            |
block5b_add[0][0] |
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block6a_expand_conv | (None, 14, 14,    |     75,264 |
block5c_add[0][0] |
| (Conv2D)            | 672)              |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block6a_expand_bn   | (None, 14, 14,    |      2,688 |
```

```
block6a_expand_c… │
│ (BatchNormalizatio… │ 672)               │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_expand_act… │ (None, 14, 14,     │          0 │
block6a_expand_b… │
│ (Activation)        │ 672)               │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_dwconv_pad  │ (None, 17, 17,     │          0 │
block6a_expand_a… │
│ (ZeroPadding2D)     │ 672)               │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_dwconv      │ (None, 7, 7, 672)  │     16,800 │
block6a_dwconv_p… │
│ (DepthwiseConv2D)   │                    │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_bn          │ (None, 7, 7, 672)  │      2,688 │
block6a_dwconv[0… │
│ (BatchNormalizatio… │                    │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_activation  │ (None, 7, 7, 672)  │          0 │ block6a_bn[0]
[0]  │
│ (Activation)        │                    │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_se_squeeze  │ (None, 672)        │          0 │
block6a_activati… │
│ (GlobalAveragePool… │                    │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_se_reshape  │ (None, 1, 1, 672)  │          0 │
block6a_se_squee… │
│ (Reshape)           │                    │            │
│                     │                    │            │
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
│ block6a_se_reduce   │ (None, 1, 1, 28)   │     18,844 │
block6a_se_resha… │
│ (Conv2D)            │                    │            │
```

```
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6a_se_expand     │ (None, 1, 1, 672)   │      19,488 │
block6a_se_reduc… │
│ (Conv2D)              │                      │             │
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6a_se_excite     │ (None, 7, 7, 672)   │           0 │
block6a_activati… │
│ (Multiply)            │                      │             │
block6a_se_expan… │
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6a_project_co… │ (None, 7, 7, 192)   │     129,024 │
block6a_se_excit… │
│ (Conv2D)              │                      │             │
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6a_project_bn   │ (None, 7, 7, 192)   │         768 │
block6a_project_… │
│ (BatchNormalizatio… │                      │             │
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6b_expand_conv │ (None, 7, 7,         │     221,184 │
block6a_project_… │
│ (Conv2D)              │ 1152)                │             │
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6b_expand_bn    │ (None, 7, 7,         │       4,608 │
block6b_expand_c… │
│ (BatchNormalizatio… │ 1152)                │             │
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6b_expand_act… │ (None, 7, 7,         │           0 │
block6b_expand_b… │
│ (Activation)          │ 1152)                │             │
│
├───────────────────────┼──────────────────────┼─────────────┼───────────────
───────┤
│ block6b_dwconv        │ (None, 7, 7,         │      28,800 │
block6b_expand_a… │
│ (DepthwiseConv2D)    │ 1152)                │             │
│
```

```
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_bn           │ (None, 7, 7,         │       4,608 │
block6b_dwconv[0… │
│ (BatchNormalizatio… │ 1152)                │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_activation   │ (None, 7, 7,         │           0 │ block6b_bn[0]
[0]  │
│ (Activation)         │ 1152)                │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_squeeze   │ (None, 1152)         │           0 │
block6b_activati… │
│ (GlobalAveragePool… │                      │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_reshape   │ (None, 1, 1,         │           0 │
block6b_se_squee… │
│ (Reshape)            │ 1152)                │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_reduce    │ (None, 1, 1, 48)     │      55,344 │
block6b_se_resha… │
│ (Conv2D)             │                      │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_expand    │ (None, 1, 1,         │      56,448 │
block6b_se_reduc… │
│ (Conv2D)             │ 1152)                │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_excite    │ (None, 7, 7,         │           0 │
block6b_activati… │
│ (Multiply)           │ 1152)                │             │
block6b_se_expan… │
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_project_co… │ (None, 7, 7, 192)    │     221,184 │
block6b_se_excit… │
│ (Conv2D)             │                      │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
```

```
──────┤
| block6b_project_bn  | (None, 7, 7, 192) |         768 |
block6b_project_… |
| (BatchNormalizatio… |                   |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6b_drop        | (None, 7, 7, 192) |           0 |
block6b_project_… |
| (Dropout)           |                   |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6b_add (Add)   | (None, 7, 7, 192) |           0 |
block6b_drop[0][… |
|                     |                   |             |
block6a_project_… |
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6c_expand_conv | (None, 7, 7,      |     221,184 |
block6b_add[0][0] |
| (Conv2D)            | 1152)             |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6c_expand_bn   | (None, 7, 7,      |       4,608 |
block6c_expand_c… |
| (BatchNormalizatio… | 1152)             |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6c_expand_act… | (None, 7, 7,      |           0 |
block6c_expand_b… |
| (Activation)        | 1152)             |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6c_dwconv      | (None, 7, 7,      |      28,800 |
block6c_expand_a… |
| (DepthwiseConv2D)   | 1152)             |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| block6c_bn          | (None, 7, 7,      |       4,608 |
block6c_dwconv[0… |
| (BatchNormalizatio… | 1152)             |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
```

block6c_activation (Activation)	(None, 7, 7, 1152)	0	block6c_bn[0][0]
block6c_se_squeeze (GlobalAveragePool…	(None, 1152)	0	block6c_activati…
block6c_se_reshape (Reshape)	(None, 1, 1, 1152)	0	block6c_se_squee…
block6c_se_reduce (Conv2D)	(None, 1, 1, 48)	55,344	block6c_se_resha…
block6c_se_expand (Conv2D)	(None, 1, 1, 1152)	56,448	block6c_se_reduc…
block6c_se_excite (Multiply)	(None, 7, 7, 1152)	0	block6c_activati… block6c_se_expan…
block6c_project_co… (Conv2D)	(None, 7, 7, 192)	221,184	block6c_se_excit…
block6c_project_bn (BatchNormalizatio…	(None, 7, 7, 192)	768	block6c_project_…
block6c_drop	(None, 7, 7, 192)	0	

```
block6c_project_… |
| (Dropout)            |                      |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6c_add (Add)    | (None, 7, 7, 192) |           0 |
block6c_drop[0][… |
|                      |                      |             |
block6b_add[0][0] |
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_expand_conv | (None, 7, 7,       |     221,184 |
block6c_add[0][0] |
| (Conv2D)             | 1152)                |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_expand_bn    | (None, 7, 7,       |       4,608 |
block6d_expand_c… |
| (BatchNormalizatio… | 1152)                |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_expand_act… | (None, 7, 7,       |           0 |
block6d_expand_b… |
| (Activation)         | 1152)                |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_dwconv       | (None, 7, 7,       |      28,800 |
block6d_expand_a… |
| (DepthwiseConv2D)    | 1152)                |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_bn           | (None, 7, 7,       |       4,608 |
block6d_dwconv[0… |
| (BatchNormalizatio… | 1152)                |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_activation   | (None, 7, 7,       |           0 | block6d_bn[0]
[0]   |
| (Activation)         | 1152)                |             |
|
├──────────────────────┼──────────────────────┼─────────────┼──────────────────
──────┤
| block6d_se_squeeze   | (None, 1152)       |           0 |
block6d_activati… |
```

(GlobalAveragePool…			
block6d_se_reshape (Reshape)	(None, 1, 1, 1152)	0	block6d_se_squee…
block6d_se_reduce (Conv2D)	(None, 1, 1, 48)	55,344	block6d_se_resha…
block6d_se_expand (Conv2D)	(None, 1, 1, 1152)	56,448	block6d_se_reduc…
block6d_se_excite (Multiply)	(None, 7, 7, 1152)	0	block6d_activati… block6d_se_expan…
block6d_project_co… (Conv2D)	(None, 7, 7, 192)	221,184	block6d_se_excit…
block6d_project_bn (BatchNormalizatio…	(None, 7, 7, 192)	768	block6d_project_…
block6d_drop (Dropout)	(None, 7, 7, 192)	0	block6d_project_…
block6d_add (Add)	(None, 7, 7, 192)	0	block6d_drop[0][…

```
block6c_add[0][0] │
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_expand_conv │ (None, 7, 7,       │    221,184 │
block6d_add[0][0] │
│ (Conv2D)             │ 1152)              │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_expand_bn    │ (None, 7, 7,       │      4,608 │
block7a_expand_c… │
│ (BatchNormalizatio… │ 1152)              │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_expand_act… │ (None, 7, 7,       │          0 │
block7a_expand_b… │
│ (Activation)         │ 1152)              │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_dwconv       │ (None, 7, 7,       │     10,368 │
block7a_expand_a… │
│ (DepthwiseConv2D)    │ 1152)              │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_bn           │ (None, 7, 7,       │      4,608 │
block7a_dwconv[0… │
│ (BatchNormalizatio… │ 1152)              │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_activation   │ (None, 7, 7,       │          0 │ block7a_bn[0]
[0]  │
│ (Activation)         │ 1152)              │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_se_squeeze   │ (None, 1152)       │          0 │
block7a_activati… │
│ (GlobalAveragePool… │                    │             │
│
├──────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
│ block7a_se_reshape   │ (None, 1, 1,       │          0 │
block7a_se_squee… │
│ (Reshape)            │ 1152)              │             │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block7a_se_reduce    │ (None, 1, 1, 48)     │       55,344 │ block7a_se_resha… │
│ (Conv2D)             │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block7a_se_expand    │ (None, 1, 1,         │       56,448 │ block7a_se_reduc… │
│ (Conv2D)             │ 1152)                │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block7a_se_excite    │ (None, 7, 7,         │            0 │ block7a_activati… │
│ (Multiply)           │ 1152)                │              │ block7a_se_expan… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block7a_project_co…  │ (None, 7, 7, 320)    │      368,640 │ block7a_se_excit… │
│ (Conv2D)             │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block7a_project_bn   │ (None, 7, 7, 320)    │        1,280 │ block7a_project_… │
│ (BatchNormalizatio…  │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ top_conv (Conv2D)    │ (None, 7, 7,         │      409,600 │ block7a_project_… │
│                      │ 1280)                │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ top_bn               │ (None, 7, 7,         │        5,120 │ top_conv[0][0]    │
│ (BatchNormalizatio…  │ 1280)                │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ top_activation       │ (None, 7, 7,         │            0 │ top_bn[0][0]      │
│ (Activation)         │ 1280)                │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
```

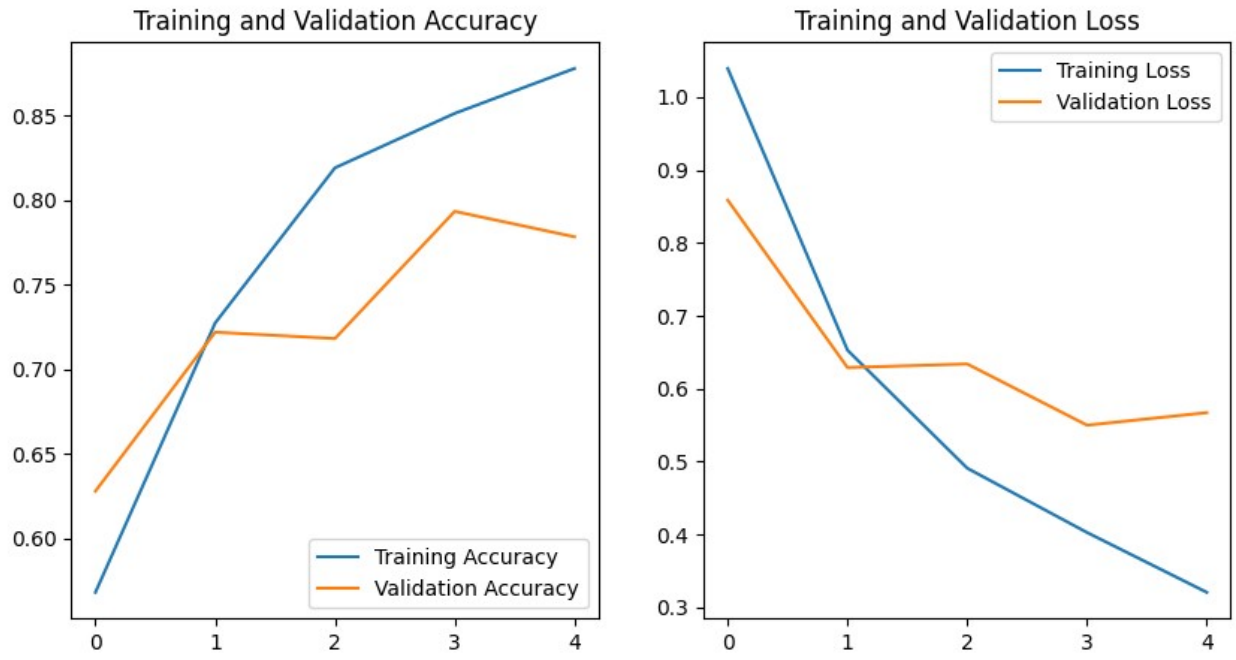
```
──────┤
| global_average_poo… | (None, 1280)      |          0 |
top_activation[0… |
| (GlobalAveragePool… |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
| dense_5 (Dense)     | (None, 1024)      |  1,311,744 |
global_average_p… |
├─────────────────────┼───────────────────┼────────────┼──────────────────
──────┤
| dense_6 (Dense)     | (None, 4)         |      4,100 | dense_5[0][0]
|
└─────────────────────┴───────────────────┴────────────┴──────────────────
──────┘

 Total params: 5,365,415 (20.47 MB)

 Trainable params: 1,315,844 (5.02 MB)

 Non-trainable params: 4,049,571 (15.45 MB)
```

```
# Compile the model
EfficientNetB0model.compile(optimizer=Adam(learning_rate=0.001),
              loss='sparse_categorical_crossentropy',  # Suitable for
multi-class classification
              metrics=['accuracy'])

# Fit the model
epochs=20
history = EfficientNetB0model.fit(train_ds,
validation_data=validation_ds, epochs=epochs, callbacks =
[early_stopping, reduce_lr] )
```

```
Epoch 1/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 93s 1s/step - accuracy: 0.4888 - loss:
1.1776 - val_accuracy: 0.6278 - val_loss: 0.8586 - learning_rate:
0.0010
Epoch 2/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 51s 1s/step - accuracy: 0.7038 - loss:
0.6989 - val_accuracy: 0.7218 - val_loss: 0.6290 - learning_rate:
0.0010
Epoch 3/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 55s 1s/step - accuracy: 0.8320 - loss:
0.4674 - val_accuracy: 0.7180 - val_loss: 0.6339 - learning_rate:
0.0010
Epoch 4/20
45/45 ━━━━━━━━━━━━━━━━━━━━ 55s 1s/step - accuracy: 0.8504 - loss:
0.4024 - val_accuracy: 0.7932 - val_loss: 0.5500 - learning_rate:
0.0010
Epoch 5/20
```

```
45/45 ━━━━━━━━━━━━━━━━━━━━ 59s 1s/step - accuracy: 0.8745 - loss:
0.3158 - val_accuracy: 0.7782 - val_loss: 0.5671 - learning_rate:
0.0010
Epoch 5: early stopping
Restoring model weights from the end of the best epoch: 1.
```

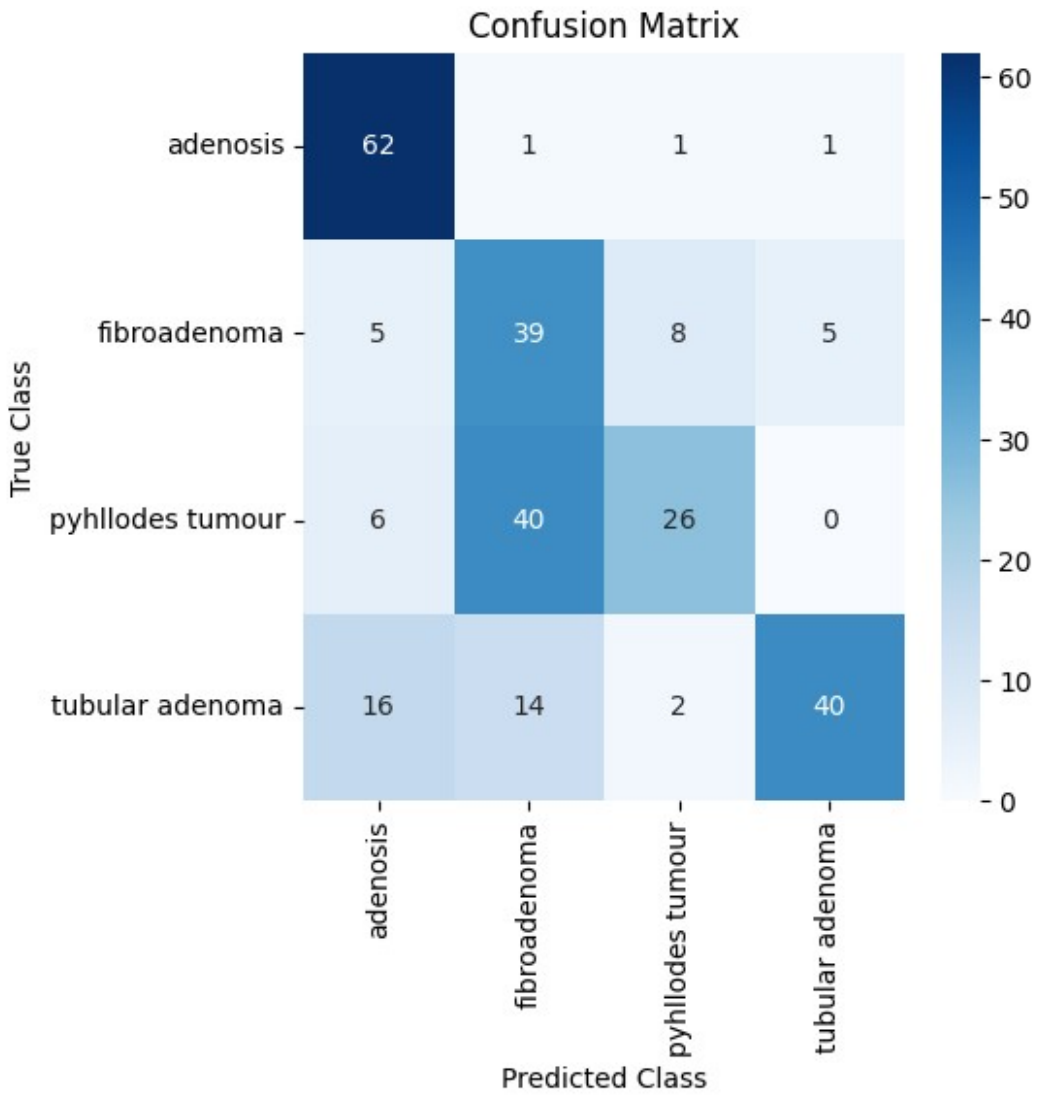
```python
EfficientNetB0_acc = history.history['accuracy']
EfficientNetB0_val_acc = history.history['val_accuracy']

EfficientNetB0_loss = history.history['loss']
EfficientNetB0_val_loss = history.history['val_loss']

epochs_range = range(len(history.history['accuracy']))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, EfficientNetB0_acc, label='Training Accuracy')
plt.plot(epochs_range, EfficientNetB0_val_acc, label='Validation
Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, EfficientNetB0_loss, label='Training Loss')
plt.plot(epochs_range, EfficientNetB0_val_loss, label='Validation
Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```



```
# Evaluate the model on the validation/test dataset
EfficientNetB0_val_loss, EfficientNetB0_val_accuracy =
EfficientNetB0model.evaluate(validation_ds)
print(f'EfficientNetB0 Validation Loss: {EfficientNetB0_val_loss}')
print(f'EfficientNetB0 Validation Accuracy:
{EfficientNetB0_val_accuracy}')

# Predict the classes on the validation/test dataset
EfficientNetB0_Y_pred = EfficientNetB0model.predict(validation_ds)
EfficientNet_y_pred = EfficientNetB0_Y_pred.argmax(axis=1)  # Get the
index of the max probability (predicted class)
```

```
10/10 ━━━━━━━━━━━━━━━━━━━━ 10s 1s/step - accuracy: 0.6309 - loss:
0.8611
EfficientNetB0 Validation Loss: 0.8586029410362244
EfficientNetB0 Validation Accuracy: 0.6278195381164551
10/10 ━━━━━━━━━━━━━━━━━━━━ 24s 1s/step
```

```
#3) EfficientNetB0
#  Check if y_true needs to be converted
if y_true.ndim == 1:
    EfficientNetB0_y_true_classes = y_true  # If it's already a vector
of class labels
else:
    EfficientNetB0_y_true_classes = np.argmax(y_true, axis=1)  # If
it's one-hot encoded

# y_pred_classes is already computed correctly
EfficientNetB0_y_pred_classes = np.argmax(EfficientNetB0_Y_pred,
axis=1)  # Predicted class labels

# Compute the confusion matrix
conf_matrix = confusion_matrix(EfficientNetB0_y_true_classes,
EfficientNetB0_y_pred_classes)

# Plot the confusion matrix
# Plot the confusion matrix
class_names = ["adenosis", "fibroadenoma", "pyhllodes tumour","tubular
adenoma"]

plt.figure(figsize=(5, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Class')
plt.ylabel('True Class')
plt.show()
```




```
# predicted probabilities from the model
y_pred_prob = EfficientNetB0model.predict(X_test)

# Step 2: Convert y_test to one-hot encoding (if not already done)
y_test_one_hot = to_categorical(y_test, num_classes = 4)

# Step 3: Calculate the ROC AUC score for multi-class classification
roc_auc = roc_auc_score(y_test_one_hot, y_pred_prob,
multi_class='ovr')
print(f"ROC AUC: {roc_auc}")
```

```
9/9 ━━━━━━━━━━━━━━━━━━━━ 21s 2s/step
ROC AUC: 0.888518287084948
```

```
# Classification Report
#3) EfficientNetB0
```

```
report = classification_report(EfficientNetB0_y_true_classes,
EfficientNetB0_y_pred_classes, target_names=class_names)
print('Classification Report:')
print(report)
```

```
Classification Report:
                  precision    recall  f1-score   support

        adenosis       0.70      0.95      0.81        65
    fibroadenoma       0.41      0.68      0.52        57
pyhllodes tumour       0.70      0.36      0.48        72
 tubular adenoma       0.87      0.56      0.68        72

        accuracy                           0.63       266
       macro avg       0.67      0.64      0.62       266
    weighted avg       0.68      0.63      0.62       266
```

```
import matplotlib.pyplot as plt
import numpy as np
import PIL
import tensorflow as tf
import pathlib
from PIL import Image
from pathlib import Path
import random
import cleanvision as cv
from cleanvision.imagelab import Imagelab
import os
import seaborn as sns
import pandas as pd

import sklearn as skl
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.utils import resample
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten, Conv2D,
MaxPooling2D, Dropout, BatchNormalization, GlobalAveragePooling2D,
Rescaling
from tensorflow.keras.models import Sequential, load_model
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.applications import ResNet50
from tensorflow.keras.applications import EfficientNetB0
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Model
from tensorflow.keras.utils import image_dataset_from_directory as
tfds
from tensorflow.keras.callbacks import EarlyStopping,
ReduceLROnPlateau
from tensorflow.keras.metrics import AUC

from typing import Tuple
```

```
c:\Users\solyl\OneDrive - University of Cape Town\Masters\Code\.venv\
Lib\site-packages\tqdm\auto.py:21: TqdmWarning: IProgress not found.
Please update jupyter and ipywidgets. See
https://ipywidgets.readthedocs.io/en/stable/user_install.html
  from .autonotebook import tqdm as notebook_tqdm
```

```
fold_info = pd.read_csv(r"C:\Users\solyl\OneDrive - University of Cape
Town\Masters\Code\breakhis\Folds.csv")
fold_info["label"] = fold_info["filename"].str.extract("(malignant)")
fold_info.head()
```

```
   fold  mag    grp                                                filename
label
0     1  100  train  BreaKHis_v1/histology_slides/breast/benign/SOB...
NaN
1     1  100  train  BreaKHis_v1/histology_slides/breast/benign/SOB...
NaN
2     1  100  train  BreaKHis_v1/histology_slides/breast/benign/SOB...
NaN
3     1  100  train  BreaKHis_v1/histology_slides/breast/benign/SOB...
NaN
4     1  100  train  BreaKHis_v1/histology_slides/breast/benign/SOB...
NaN
```

```
# Loading the dataset - looking at only MALIGNANT
# Convert the string path to a Path object"C:/Users/solyl/OneDrive -
University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/malignant/SOB")
data_dir = pathlib.Path("C:/Users/solyl/OneDrive - University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/malignant/SOB")

# Use glob to count the number of MALIGNANT .png files
image_count = len(list(data_dir.glob('**/*.png')))
print(image_count)

ductal_carcinoma_dir = pathlib.Path("C:/Users/solyl/OneDrive -
University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/malignant/SOB/ductal_carcinoma")
ductal_carcinoma_count =
len(list(ductal_carcinoma_dir.glob('**/*.png')))
print(ductal_carcinoma_count)

lobular_carcinoma_dir = pathlib.Path("C:/Users/solyl/OneDrive -
University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/malignant/SOB/lobular_carcinoma")
lobular_carcinoma_count =
len(list(lobular_carcinoma_dir.glob('**/*.png')))
print(lobular_carcinoma_count)

mucinous_carcinoma_dir = pathlib.Path("C:/Users/solyl/OneDrive -
University of Cape
Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/
breast/malignant/SOB/mucinous_carcinoma")
mucinous_carcinoma_count =
len(list(mucinous_carcinoma_dir.glob('**/*.png')))
print(mucinous_carcinoma_count)
```

```
papillary_carcinoma_dir = pathlib.Path("C:/Users/solyl/OneDrive - University of Cape Town/Masters/Code/Dataset/BreaKHis_v1/BreaKHis_v1/histology_slides/breast/malignant/SOB/papillary_carcinoma")
papillary_carcinoma_count = len(list(papillary_carcinoma_dir.glob('**/*.png')))
print(papillary_carcinoma_count)
```

```
2240
560
560
560
560
```

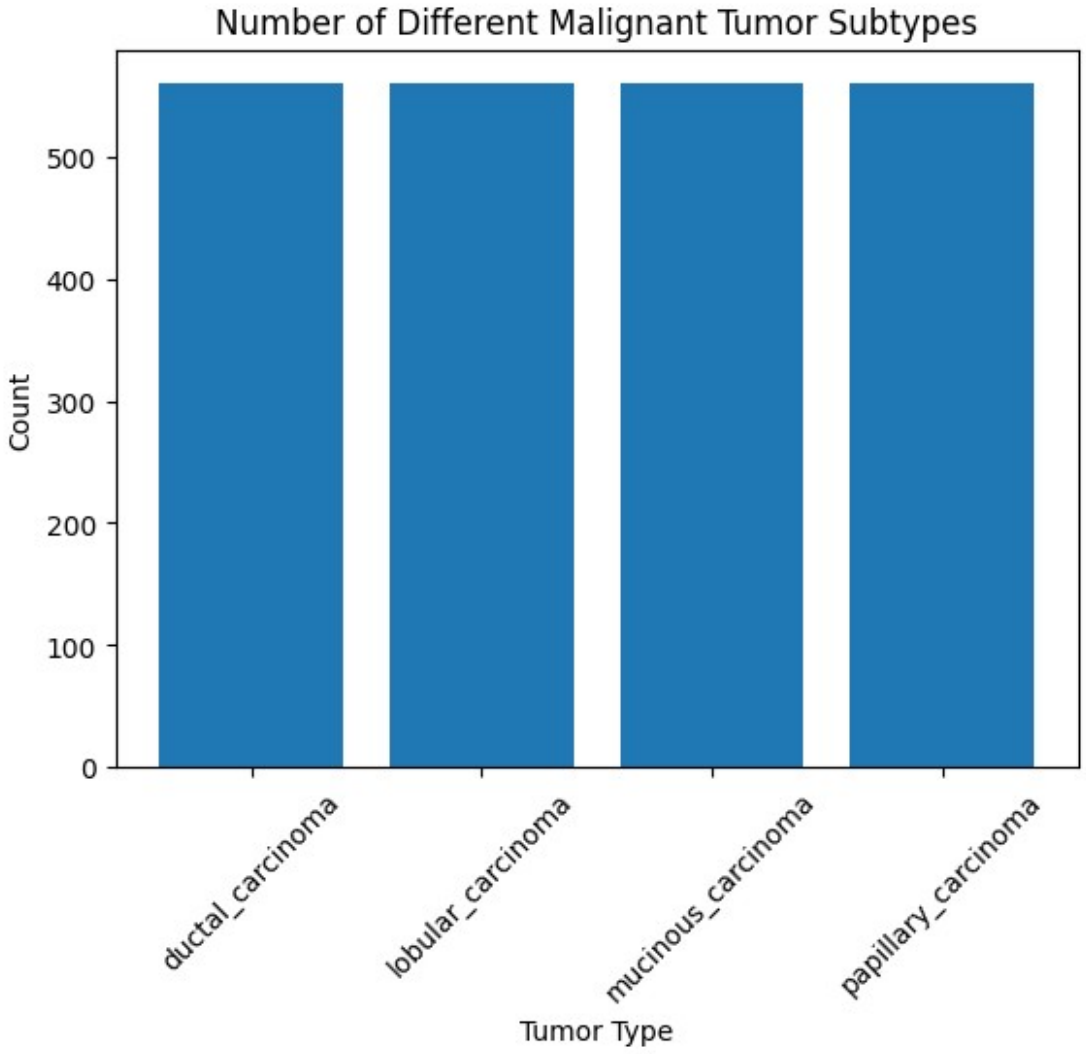
```
# Initialize an empty dictionary to store the file counts for each folder.
class_counts = {}

# Path to the main directory containing subfolders.
main_dir = data_dir
min_images = 560

# Loop through each folder in the main directory.
for class_name in os.listdir(main_dir):
    class_dir = os.path.join(main_dir, class_name)

    # Check if the item is a directory (to exclude files in the main_dir).
    if os.path.isdir(class_dir):
        # Initialize a file count for this folder.
        file_count = 0
        images = []

        # Walk through all subfolders in the class_dir to count files and collect file paths.
        for root, dirs, files in os.walk(class_dir):
            for file in files:
                # Ensure we're only collecting files (and filtering images, if needed)
                file_path = os.path.join(root, file)
                if os.path.isfile(file_path):  # Make sure it's a file
                    images.append(file_path)
                    file_count += 1

        # Store the count in the dictionary with the class_name as the key.
        class_counts[class_name] = file_count

        # Debugging output to ensure we counted the correct number of files
```

```
        print(f"{class_name} has {file_count} images.")

        # Now, delete images if there are more than min_images in this
class
        if file_count > min_images:
            excess_images = file_count - min_images
            print(f"{class_name} has {excess_images} excess images.
Preparing to delete...")

            # Randomly select images to delete (ensure we don't delete
more than we have)
            if excess_images > 0 and excess_images <= len(images):
                images_to_delete = random.sample(images,
excess_images)

                # Delete the selected images
                for image_path in images_to_delete:
                    print(f"Attempting to delete: {image_path}")  #
Debugging statement
                    try:
                        os.remove(image_path)
                        print(f"Deleted {image_path} from
{class_name}")
                    except Exception as e:
                        print(f"Error deleting {image_path}: {e}")
            else:
                print(f"Skipping deletion for {class_name}. No excess
images to delete.")
        else:
            print(f"No excess images to delete for {class_name}.")
    else:
        print(f"Skipping {class_name}, as it is not a directory.")

# Output the dictionary with counts after deletions
print("Final class counts after deletion:")
print(class_counts)
```

```
ductal_carcinoma has 560 images.
No excess images to delete for ductal_carcinoma.
lobular_carcinoma has 560 images.
No excess images to delete for lobular_carcinoma.
mucinous_carcinoma has 560 images.
No excess images to delete for mucinous_carcinoma.
papillary_carcinoma has 560 images.
No excess images to delete for papillary_carcinoma.
Final class counts after deletion:
{'ductal_carcinoma': 560, 'lobular_carcinoma': 560,
'mucinous_carcinoma': 560, 'papillary_carcinoma': 560}
```

```
names = list(class_counts.keys())
counts = list(class_counts.values())


plt.bar(names, counts)
plt.title('Number of Different Malignant Tumor Subtypes')
plt.xlabel('Tumor Type')
plt.ylabel('Count')
plt.xticks(rotation=45)
```

```
([0, 1, 2, 3],
 [Text(0, 0, 'ductal_carcinoma'),
  Text(1, 0, 'lobular_carcinoma'),
  Text(2, 0, 'mucinous_carcinoma'),
  Text(3, 0, 'papillary_carcinoma')])
```



```python
# Image Validation- using Imagelab from cleanvision to detect issues
in the image dataset
# DUCTAL CARCINOMA

ductal_carcinoma_images = list(ductal_carcinoma_dir.glob('**/*.png'))

ductal_carcinoma_filepaths = [str(path) for path in
ductal_carcinoma_images]  # Convert WindowsPath objects to strings

# Applying ImageLab
imagelab = Imagelab(filepaths=ductal_carcinoma_filepaths)
imagelab.find_issues()
imagelab.report()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 560/560 [00:16<00:00, 34.96it/s]
100%|██████████| 560/560 [00:09<00:00, 61.78it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset
```

	issue_type	num_images
0	dark	0
1	light	0
2	odd_aspect_ratio	0
3	low_information	0
4	blurry	0
5	grayscale	0
6	odd_size	0
7	exact_duplicates	0
8	near_duplicates	0

```python
# Filter images flagged as exact or near duplicates
duplicated_images = imagelab.issues[
    imagelab.issues[['is_exact_duplicates_issue',
'is_near_duplicates_issue']].any(axis=1)
].reset_index().rename(columns={'index': 'filename'})

# Display the filtered DataFrame to check which images are duplicates
print(duplicated_images.head())
```

```
Empty DataFrame
Columns: [filename, odd_size_score, is_odd_size_issue,
```

```
odd_aspect_ratio_score, is_odd_aspect_ratio_issue,
low_information_score, is_low_information_issue, light_score,
is_light_issue, grayscale_score, is_grayscale_issue, dark_score,
is_dark_issue, blurry_score, is_blurry_issue, exact_duplicates_score,
is_exact_duplicates_issue, near_duplicates_score,
is_near_duplicates_issue]
Index: []
```

```
# Convert the filenames to a set for fast lookup
duplicate_files = set(duplicated_images['filename'])

# Remove duplicate images from the original filepaths list
cleaned_ductal_carcinoma_filepaths = [f for f in
ductal_carcinoma_filepaths if f not in duplicate_files]

# Display the number of files before and after removing duplicates
print(f"Original number of files: {len(ductal_carcinoma_filepaths)}")
print(f"Number of files after removing duplicates:
{len(cleaned_ductal_carcinoma_filepaths)}")
```

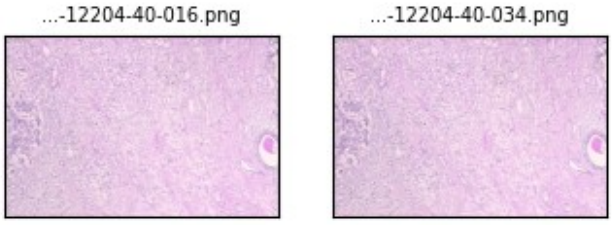
```
Original number of files: 560
Number of files after removing duplicates: 560
```

```
# Validate again
imagelab = Imagelab(filepaths=cleaned_ductal_carcinoma_filepaths)
imagelab.find_issues()
```

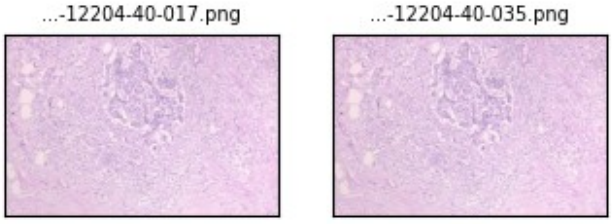
```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 560/560 [00:15<00:00, 35.78it/s]
100%|██████████| 560/560 [00:13<00:00, 40.99it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
```

```
# Image Validation- using Imagelab from cleanvision to detect issues
in the image dataset
# LOBULAR CARCINOMA

lobular_carcinoma_images =
list(lobular_carcinoma_dir.glob('**/*.png'))

lobular_carcinoma_filepaths = [str(path) for path in
lobular_carcinoma_images]  # Convert WindowsPath objects to strings

# Applying ImageLab
imagelab = Imagelab(filepaths=lobular_carcinoma_filepaths)
```

```
imagelab.find_issues()
imagelab.report()

Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 560/560 [00:15<00:00, 36.74it/s]
100%|██████████| 560/560 [00:19<00:00, 28.47it/s]

Issue checks completed. 4 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | exact_duplicates |            4 |
|  1 | dark             |            0 |
|  2 | light            |            0 |
|  3 | odd_aspect_ratio |            0 |
|  4 | low_information  |            0 |
|  5 | blurry           |            0 |
|  6 | grayscale        |            0 |
|  7 | odd_size         |            0 |
|  8 | near_duplicates  |            0 |

----------------- exact_duplicates images ------------------

Number of examples with this issue: 4
Examples representing most severe instances of this issue:

Set: 0
```




```
Set: 1
```



```python
# Filter images flagged as exact or near duplicates
duplicated_images = imagelab.issues[
    imagelab.issues[['is_exact_duplicates_issue',
'is_near_duplicates_issue']].any(axis=1)
].reset_index().rename(columns={'index': 'filename'})

# Display the filtered DataFrame to check which images are duplicates
print(duplicated_images.head())
```

```
                                            filename
odd_size_score  \
0  C:\Users\solyl\OneDrive - University of Cape T...             1.0

1  C:\Users\solyl\OneDrive - University of Cape T...             1.0

2  C:\Users\solyl\OneDrive - University of Cape T...             1.0

3  C:\Users\solyl\OneDrive - University of Cape T...             1.0


   is_odd_size_issue  odd_aspect_ratio_score
is_odd_aspect_ratio_issue  \
0              False                0.657143
False
1              False                0.657143
False
2              False                0.657143
False
3              False                0.657143
False

   low_information_score  is_low_information_issue  light_score  \
0               0.749977                     False     0.289757
1               0.751213                     False     0.284295
2               0.749977                     False     0.289757
3               0.751213                     False     0.284295

   is_light_issue  grayscale_score  is_grayscale_issue  dark_score  \
0           False                1               False    0.946498
1           False                1               False    0.951503
2           False                1               False    0.946498
3           False                1               False    0.951503

   is_dark_issue  blurry_score  is_blurry_issue
exact_duplicates_score  \
0          False      0.687215            False
0.5
1          False      0.701148            False
0.5
2          False      0.687215            False
```

```
0.5
3          False      0.701148            False
0.5

   is_exact_duplicates_issue  near_duplicates_score
is_near_duplicates_issue
0                       True                    1.0
False
1                       True                    1.0
False
2                       True                    1.0
False
3                       True                    1.0
False
```

```python
# Convert the filenames to a set for fast lookup
duplicate_files = set(duplicated_images['filename'])

# Remove duplicate images from the original filepaths list
cleaned_lobular_carcinoma_filepaths = [f for f in
lobular_carcinoma_filepaths if f not in duplicate_files]

# Display the number of files before and after removing duplicates
print(f"Original number of files: {len(lobular_carcinoma_filepaths)}")
print(f"Number of files after removing duplicates:
{len(cleaned_lobular_carcinoma_filepaths)}")
```

```
Original number of files: 560
Number of files after removing duplicates: 556
```

```python
# Validate again
imagelab = Imagelab(filepaths=cleaned_lobular_carcinoma_filepaths)
imagelab.find_issues()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 556/556 [00:16<00:00, 32.87it/s]
100%|██████████| 556/556 [00:09<00:00, 59.05it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
```

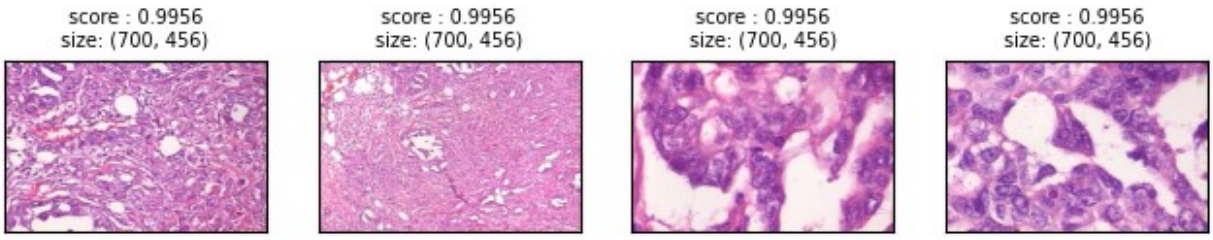
```python
# Image Validation- using Imagelab from cleanvision to detect issues
in the image dataset
# MUCINOUS CARCINOMA

mucinous_carcinoma_images =
list(mucinous_carcinoma_dir.glob('**/*.png'))
```

```
mucinous_carcinoma_filepaths = [str(path) for path in
mucinous_carcinoma_images]  # Convert WindowsPath objects to strings

# Applying ImageLab
imagelab = Imagelab(filepaths=mucinous_carcinoma_filepaths)
imagelab.find_issues()
imagelab.report()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 560/560 [00:14<00:00, 39.00it/s]
100%|██████████| 560/560 [00:10<00:00, 53.45it/s]

Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | dark             |            0 |
|  1 | light            |            0 |
|  2 | odd_aspect_ratio |            0 |
|  3 | low_information  |            0 |
|  4 | blurry           |            0 |
|  5 | grayscale        |            0 |
|  6 | odd_size         |            0 |
|  7 | exact_duplicates |            0 |
|  8 | near_duplicates  |            0 |
```

```
# Image Validation- using Imagelab from cleanvision to detect issues
in the image dataset
# PAPILLARY CARCINOMA

papillary_carcinoma_images =
list(papillary_carcinoma_dir.glob('**/*.png'))

papillary_carcinoma_filepaths = [str(path) for path in
papillary_carcinoma_images]  # Convert WindowsPath objects to strings

# Applying ImageLab
imagelab = Imagelab(filepaths=papillary_carcinoma_filepaths)
imagelab.find_issues()
imagelab.report()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 560/560 [00:13<00:00, 40.97it/s]
100%|██████████| 560/560 [00:08<00:00, 63.01it/s]

Issue checks completed. 74 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
Issues found in images in order of severity in the dataset

|    | issue_type       |   num_images |
|---:|:-----------------|-------------:|
|  0 | odd_size         |           74 |
|  1 | dark             |            0 |
|  2 | light            |            0 |
|  3 | odd_aspect_ratio |            0 |
|  4 | low_information  |            0 |
|  5 | blurry           |            0 |
|  6 | grayscale        |            0 |
|  7 | exact_duplicates |            0 |
|  8 | near_duplicates  |            0 |

---------------------- odd_size images ----------------------

Number of examples with this issue: 74
Examples representing most severe instances of this issue:
```




```python
# Filter images flagged as odd sizes
odd_size_images = imagelab.issues[
    imagelab.issues[['is_odd_size_issue']].any(axis=1)
].reset_index().rename(columns={'index': 'filename'})

# Display the filtered DataFrame to check which images are odd sizes
print(odd_size_images)
```

```
                                            filename  odd_size_score
\
0   C:\Users\solyl\OneDrive - University of Cape T...        0.995643

1   C:\Users\solyl\OneDrive - University of Cape T...        0.995643
```

```
2   C:\Users\solyl\OneDrive - University of Cape T...        0.995643

3   C:\Users\solyl\OneDrive - University of Cape T...        0.995643

4   C:\Users\solyl\OneDrive - University of Cape T...        0.995643

..                                                  ...             ...

69  C:\Users\solyl\OneDrive - University of Cape T...        0.995643

70  C:\Users\solyl\OneDrive - University of Cape T...        0.995643

71  C:\Users\solyl\OneDrive - University of Cape T...        0.995643

72  C:\Users\solyl\OneDrive - University of Cape T...        0.995643

73  C:\Users\solyl\OneDrive - University of Cape T...        0.995643


    is_odd_size_issue  odd_aspect_ratio_score
is_odd_aspect_ratio_issue  \
0                True                0.651429
False
1                True                0.651429
False
2                True                0.651429
False
3                True                0.651429
False
4                True                0.651429
False
..                ...                     ...                          .
..
69               True                0.651429
False
70               True                0.651429
False
71               True                0.651429
False
72               True                0.651429
False
73               True                0.651429
False

    low_information_score  is_low_information_issue  light_score  \
0                0.866140                     False     0.609940
1                0.840533                     False     0.578709
2                0.853586                     False     0.592604
3                0.848199                     False     0.550444
```

```
4                0.836051                      False     0.541288
..                    ...                        ...          ...
69               0.806630                      False     0.482967
70               0.813545                      False     0.502088
71               0.826276                      False     0.532087
72               0.818400                      False     0.474723
73               0.846071                      False     0.554166

    is_light_issue  grayscale_score  is_grayscale_issue  dark_score  \
0            False                1               False    0.974126
1            False                1               False    0.997177
2            False                1               False    0.977192
3            False                1               False    0.963151
4            False                1               False    0.983801
..             ...              ...                 ...         ...
69           False                1               False    0.995402
70           False                1               False    0.997292
71           False                1               False    0.994587
72           False                1               False    0.975753
73           False                1               False    0.982458

    is_dark_issue  blurry_score  is_blurry_issue
exact_duplicates_score  \
0           False      0.571718            False
1.0
1           False      0.556547            False
1.0
2           False      0.561198            False
1.0
3           False      0.580742            False
1.0
4           False      0.566827            False
1.0
..            ...           ...              ...                      .
..
69          False      0.689594            False
1.0
70          False      0.532045            False
1.0
71          False      0.524372            False
1.0
72          False      0.663431            False
1.0
73          False      0.718843            False
1.0

    is_exact_duplicates_issue  near_duplicates_score
is_near_duplicates_issue
0                       False                    1.0
False
```

```
1                         False                     1.0
False
2                         False                     1.0
False
3                         False                     1.0
False
4                         False                     1.0
False
..                          ...                     ...
...
69                        False                     1.0
False
70                        False                     1.0
False
71                        False                     1.0
False
72                        False                     1.0
False
73                        False                     1.0
False

[74 rows x 19 columns]
```

```
# Convert the filenames to a set for fast lookup
odd_size_files = set(odd_size_images['filename'])

# Remove odd sized images from the original filepaths list
cleaned_papillary_carcinoma_filepaths = [f for f in
papillary_carcinoma_filepaths if f not in odd_size_files]

# Display the number of files before and after removing odd sizes
print(f"Original number of files:
{len(papillary_carcinoma_filepaths)}")
print(f"Number of files after removing odd sizes:
{len(cleaned_papillary_carcinoma_filepaths)}")
```

```
Original number of files: 560
Number of files after removing odd sizes: 486
```

```
# Validate again
imagelab = Imagelab(filepaths=cleaned_papillary_carcinoma_filepaths)
imagelab.find_issues()
```

```
Checking for dark, light, odd_aspect_ratio, low_information,
exact_duplicates, near_duplicates, blurry, grayscale, odd_size
images ...

100%|██████████| 486/486 [00:29<00:00, 16.36it/s]
100%|██████████| 486/486 [00:06<00:00, 70.98it/s]
```

```
Issue checks completed. 0 issues found in the dataset. To see a
detailed report of issues found, use imagelab.report().
```

```
batch_size = 28
img_height = 224
img_width = 224

# Load the entire dataset from the directory
dataset = tf.keras.preprocessing.image_dataset_from_directory(
    data_dir,
    image_size=(img_height, img_width),
    batch_size=batch_size,
    shuffle=True
)

# Extract images and labels from the dataset
image_list = []
label_list = []

for images, labels in dataset:
    image_list.append(images.numpy())
    label_list.append(labels.numpy())

# Convert lists to NumPy arrays
X = np.concatenate(image_list)
y = np.concatenate(label_list)

# Split data into train (70%), validation (15%), and test (15%)
X_train, X_temp, y_train, y_temp = train_test_split(X, y,
test_size=0.3, random_state=42)
X_val, X_test, y_val, y_test = train_test_split(X_temp, y_temp,
test_size=0.5, random_state=42)

# Create TensorFlow datasets for train, validation, and test sets
train_ds = tf.data.Dataset.from_tensor_slices((X_train,
y_train)).batch(batch_size)
validation_ds = tf.data.Dataset.from_tensor_slices((X_val,
y_val)).batch(batch_size)
test_ds = tf.data.Dataset.from_tensor_slices((X_test,
y_test)).batch(batch_size)

# Now you have train_dataset, validation_dataset, and test_dataset
```

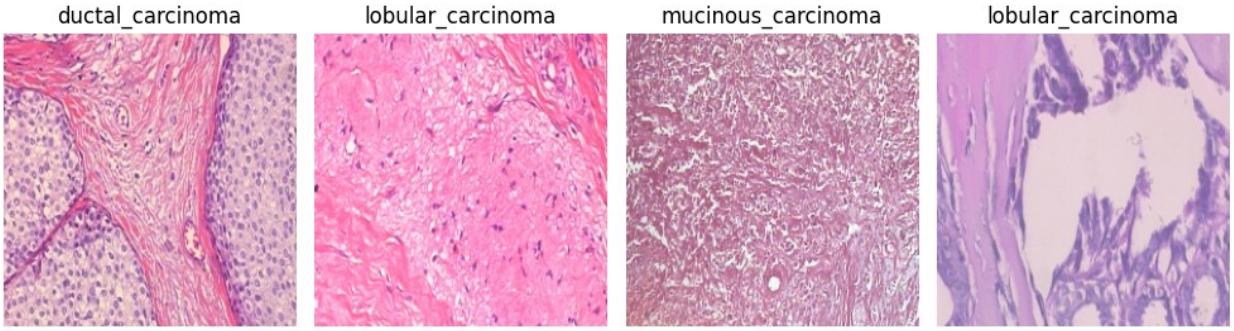
```
Found 2240 files belonging to 4 classes.
```

```
label_mapping = {0: 'ductal_carcinoma', 1: 'lobular_carcinoma', 2:
'mucinous_carcinoma', 3: 'lobular_carcinoma'}
labels = [0, 1, 2,3]  # Numeric labels
```

```
class_names = [label_mapping[label] for label in np.unique(labels)]
print("Class names:", class_names)
```

```
Class names: ['ductal_carcinoma', 'lobular_carcinoma',
'mucinous_carcinoma', 'lobular_carcinoma']
```

```
# Extract true labels
y_true = np.concatenate([y.numpy() for x, y in validation_ds], axis=0)

# Print first 9 images and labels from the training set
unique_classes = np.unique(y)

plt.figure(figsize=(10, 10))

# Iterate over each class
for i, class_label in enumerate(unique_classes):
        # Find the first image in the dataset that corresponds to this
class
        class_index = np.where(y == class_label)[0][0]
        image = X[class_index]

        # Plot the image
        plt.subplot(1, len(unique_classes), i + 1)
        plt.imshow(image.astype('uint8'))
        plt.title(class_names[i])
        plt.axis('off')

plt.tight_layout()
plt.show()
```




```
for image_batch, labels_batch in train_ds:
  print(image_batch.shape)
  print(labels_batch.shape)
  break
```

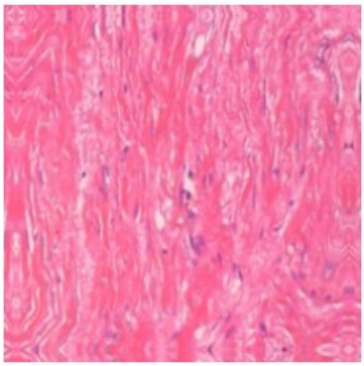
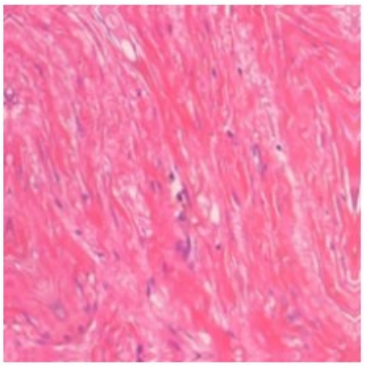
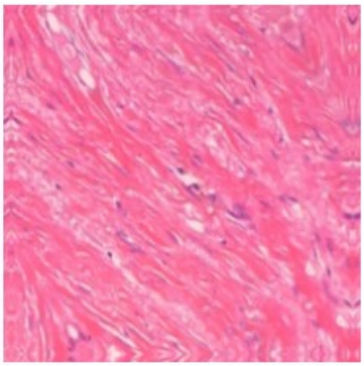
```
(28, 224, 224, 3)
(28,)
```

```
AUTOTUNE = tf.data.AUTOTUNE

train_ds =
train_ds.cache().shuffle(1000).prefetch(buffer_size=AUTOTUNE)
validation_ds = validation_ds.cache().prefetch(buffer_size=AUTOTUNE)
test_ds = test_ds.cache().prefetch(buffer_size=AUTOTUNE)

# Normalisation Layer
normalization_layer = layers.Rescaling(1./255)

normalized_ds = train_ds.map(lambda x, y: (normalization_layer(x), y))
image_batch, labels_batch = next(iter(normalized_ds))
first_image = image_batch[0]
# Pixel values are now in `[0,1]`.
print(np.min(first_image), np.max(first_image))
```

```
0.32536545 1.0
```

```
# Data Augmentation
data_augmentation = keras.Sequential(
  [
    layers.RandomFlip("horizontal",
                      input_shape=(img_height,
                                   img_width,
                                   3)),
    layers.RandomRotation(0.1),
    layers.RandomZoom(0.1),
  ]
)# View some of these
plt.figure(figsize=(10, 10))
for images, _ in train_ds.take(1):
  for i in range(3):
    augmented_images = data_augmentation(images)
    ax = plt.subplot(3, 3, i + 1)
    plt.imshow(augmented_images[0].numpy().astype("uint8"))
    plt.axis("off")
```

```
c:\Users\solyl\OneDrive - University of Cape Town\Masters\Code\.venv\
Lib\site-packages\keras\src\layers\preprocessing\tf_data_layer.py:18:
UserWarning: Do not pass an `input_shape`/`input_dim` argument to a
layer. When using Sequential models, prefer using an `Input(shape)`
object as the first layer in the model instead.
  super().__init__(**kwargs)
```

```
custom_model = tf.keras.Sequential([
    layers.Input(shape=(224, 224, 3)),
    # Data augmentation
    data_augmentation,
    # Convolutional block I
    layers.Rescaling(1./255),
    layers.BatchNormalization(),
    layers.Conv2D(32, 3, activation="relu"),
    layers.MaxPooling2D(),
    # Convolutional block II
    layers.Conv2D(64, 3, activation="relu"),
    layers.MaxPooling2D(),
    # Convolutional block III
    layers.Conv2D(128, 3, activation="relu"),
    layers.MaxPooling2D(),
    # Fully connected layers
    layers.GlobalAveragePooling2D(),
    layers.Dropout(0.4),
    layers.Dense(256, activation="relu"),
    layers.Dropout(0.3),
    layers.Dense(32, activation="relu"),
    layers.Dropout(0.2),
    layers.Dense(4, activation="softmax")
], name="CustomCNN")
custom_model.summary()
```

```
Model: "CustomCNN"
```

```
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━
━━┓
┃ Layer (type)                         ┃ Output Shape                ┃
Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━
━━┩
│ sequential (Sequential)              │ (None, 224, 224, 3)         │
0 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────
──┘
```

```
| rescaling_1 (Rescaling)          | (None, 224, 224, 3)    |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| batch_normalization              | (None, 224, 224, 3)    |
12 |
| (BatchNormalization)             |                        |
|
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| conv2d (Conv2D)                  | (None, 222, 222, 32)   |
896 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| max_pooling2d (MaxPooling2D)     | (None, 111, 111, 32)   |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| conv2d_1 (Conv2D)                | (None, 109, 109, 64)   |
18,496 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| max_pooling2d_1 (MaxPooling2D)   | (None, 54, 54, 64)     |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| conv2d_2 (Conv2D)                | (None, 52, 52, 128)    |
73,856 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| max_pooling2d_2 (MaxPooling2D)   | (None, 26, 26, 128)    |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| global_average_pooling2d         | (None, 128)            |
0 |
| (GlobalAveragePooling2D)         |                        |
|
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| dropout (Dropout)                | (None, 128)            |
0 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| dense (Dense)                    | (None, 256)            |
33,024 |
├──────────────────────────────────┼────────────────────────┼──────
──────┤
| dropout_1 (Dropout)              | (None, 256)            |
```

```
0 │
├─────────────────────────────────────┼─────────────────────────────┼──────────
──────┤
│ dense_1 (Dense)                     │ (None, 32)                  │
8,224 │
├─────────────────────────────────────┼─────────────────────────────┼──────────
──────┤
│ dropout_2 (Dropout)                 │ (None, 32)                  │
0 │
├─────────────────────────────────────┼─────────────────────────────┼──────────
──────┤
│ dense_2 (Dense)                     │ (None, 4)                   │
132 │
└─────────────────────────────────────┴─────────────────────────────┴──────────
──────┘

 Total params: 134,640 (525.94 KB)

 Trainable params: 134,634 (525.91 KB)

 Non-trainable params: 6 (24.00 B)
```

```
# Compile the model
custom_model.compile(optimizer=Adam(learning_rate=0.001),
              loss='sparse_categorical_crossentropy',  # Best to use
sparse categorical crossentropy for multi-class classification
              metrics=["accuracy"])

early_stopping = EarlyStopping(min_delta=1e-4, patience=5,
verbose=1,restore_best_weights=True)
reduce_lr = ReduceLROnPlateau(factor=0.5, patience=4, verbose=1)

# Fit the model
epochs=20
history = custom_model.fit(train_ds, validation_data=validation_ds,
epochs=epochs, callbacks = [early_stopping, reduce_lr])
```

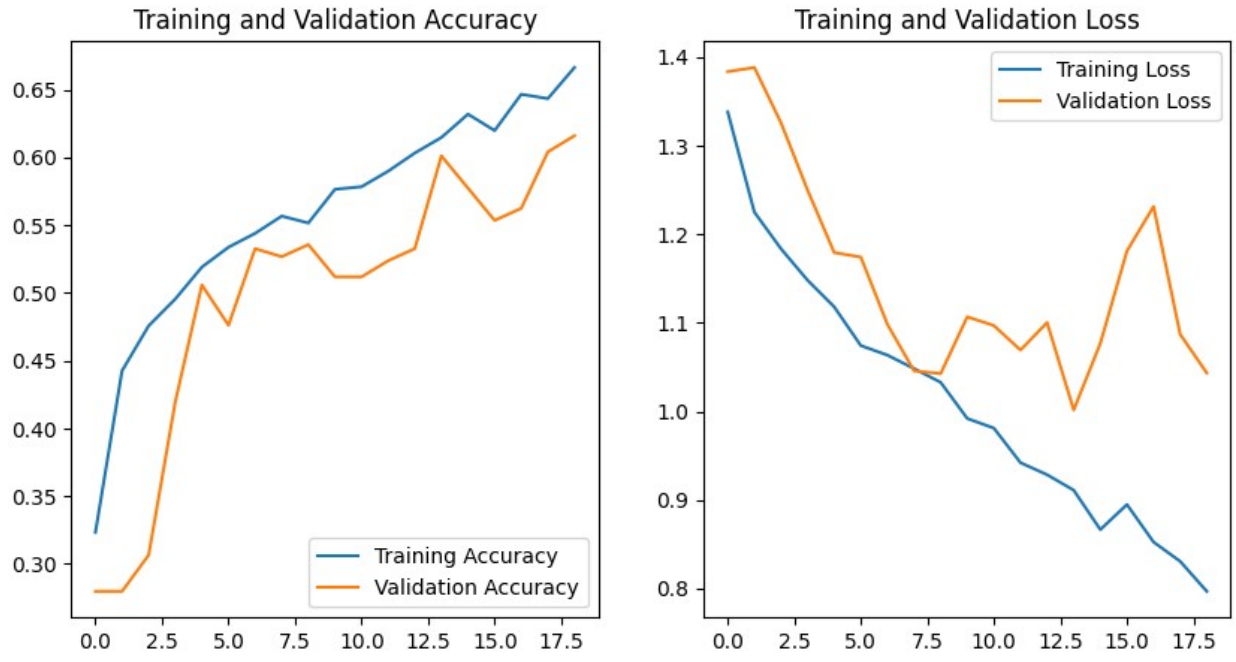
```
Epoch 1/20
32/56 ━━━━━━━━━━━━━━━━━━━━ 22s 952ms/step - accuracy: 0.2850 - loss:
1.3862

---------------------------------------------------------------------------
-----
KeyboardInterrupt                         Traceback (most recent call
last)
Cell In[32], line 3
      1 # Fit the model
      2 epochs=20
----> 3 history = custom_model.fit(train_ds,
validation_data=validation_ds, epochs=epochs, callbacks =
[early_stopping, reduce_lr])
```

```
File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\keras\src\utils\traceback_utils.py:117,
in filter_traceback.<locals>.error_handler(*args, **kwargs)
    115 filtered_tb = None
    116 try:
--> 117     return fn(*args, **kwargs)
    118 except Exception as e:
    119     filtered_tb = _process_traceback_frames(e.__traceback__)

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\keras\src\backend\tensorflow\
trainer.py:314, in TensorFlowTrainer.fit(self, x, y, batch_size,
epochs, verbose, callbacks, validation_split, validation_data,
shuffle, class_weight, sample_weight, initial_epoch, steps_per_epoch,
validation_steps, validation_batch_size, validation_freq)
    312 for step, iterator in epoch_iterator.enumerate_epoch():
    313     callbacks.on_train_batch_begin(step)
--> 314     logs = self.train_function(iterator)
    315     logs = self._pythonify_logs(logs)
    316     callbacks.on_train_batch_end(step, logs)

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\util\
traceback_utils.py:150, in
filter_traceback.<locals>.error_handler(*args, **kwargs)
    148 filtered_tb = None
    149 try:
--> 150   return fn(*args, **kwargs)
    151 except Exception as e:
    152   filtered_tb = _process_traceback_frames(e.__traceback__)

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\
polymorphic_function\polymorphic_function.py:833, in
Function.__call__(self, *args, **kwds)
    830 compiler = "xla" if self._jit_compile else "nonXla"
    832 with OptionalXlaContext(self._jit_compile):
--> 833   result = self._call(*args, **kwds)
    835 new_tracing_count = self.experimental_get_tracing_count()
    836 without_tracing = (tracing_count == new_tracing_count)

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\
polymorphic_function\polymorphic_function.py:878, in
Function._call(self, *args, **kwds)
    875 self._lock.release()
    876 # In this case we have not created variables on the first
call. So we can
    877 # run the first trace but we should fail if variables are
```

```
created.
--> 878 results = tracing_compilation.call_function(
    879     args, kwds, self._variable_creation_config
    880 )
    881 if self._created_variables:
    882   raise ValueError("Creating variables on a non-first call to
a function"
    883                    " decorated with tf.function.")

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\
polymorphic_function\tracing_compilation.py:139, in
call_function(args, kwargs, tracing_options)
    137 bound_args = function.function_type.bind(*args, **kwargs)
    138 flat_inputs = function.function_type.unpack_inputs(bound_args)
--> 139 return function._call_flat(  # pylint: disable=protected-
access
    140     flat_inputs, captured_inputs=function.captured_inputs
    141 )

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\
polymorphic_function\concrete_function.py:1322, in
ConcreteFunction._call_flat(self, tensor_inputs, captured_inputs)
   1318 possible_gradient_type =
gradients_util.PossibleTapeGradientTypes(args)
   1319 if (possible_gradient_type ==
gradients_util.POSSIBLE_GRADIENT_TYPES_NONE
   1320     and executing_eagerly):
   1321   # No tape is watching; skip to running the function.
-> 1322   return self._inference_function.call_preflattened(args)
   1323 forward_backward =
self._select_forward_and_backward_functions(
   1324     args,
   1325     possible_gradient_type,
   1326     executing_eagerly)
   1327 forward_function, args_with_tangents =
forward_backward.forward()

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\
polymorphic_function\atomic_function.py:216, in
AtomicFunction.call_preflattened(self, args)
    214 def call_preflattened(self, args: Sequence[core.Tensor]) ->
Any:
    215   """Calls with flattened tensor inputs and returns the
structured output."""
--> 216   flat_outputs = self.call_flat(*args)
    217   return self.function_type.pack_output(flat_outputs)
```

```
File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\
polymorphic_function\atomic_function.py:251, in
AtomicFunction.call_flat(self, *args)
    249 with record.stop_recording():
    250   if self._bound_context.executing_eagerly():
--> 251     outputs = self._bound_context.call_function(
    252         self.name,
    253         list(args),
    254         len(self.function_type.flat_outputs),
    255     )
    256   else:
    257     outputs = make_call_op_in_graph(
    258         self,
    259         list(args),
    260         self._bound_context.function_call_options.as_attrs(),
    261     )

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\context.py:1500,
in Context.call_function(self, name, tensor_inputs, num_outputs)
   1498 cancellation_context = cancellation.context()
   1499 if cancellation_context is None:
-> 1500   outputs = execute.execute(
   1501       name.decode("utf-8"),
   1502       num_outputs=num_outputs,
   1503       inputs=tensor_inputs,
   1504       attrs=attrs,
   1505       ctx=self,
   1506   )
   1507 else:
   1508   outputs = execute.execute_with_cancellation(
   1509       name.decode("utf-8"),
   1510       num_outputs=num_outputs,
   (...)
   1514       cancellation_manager=cancellation_context,
   1515   )

File c:\Users\solyl\OneDrive - University of Cape Town\Masters\
Code\.venv\Lib\site-packages\tensorflow\python\eager\execute.py:53, in
quick_execute(op_name, num_outputs, inputs, attrs, ctx, name)
     51 try:
     52   ctx.ensure_initialized()
---> 53   tensors = pywrap_tfe.TFE_Py_Execute(ctx._handle,
device_name, op_name,
     54                                       inputs, attrs,
num_outputs)
     55 except core._NotOkStatusException as e:
     56   if name is not None:
```

```
KeyboardInterrupt:

acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(len(history.history['accuracy']))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

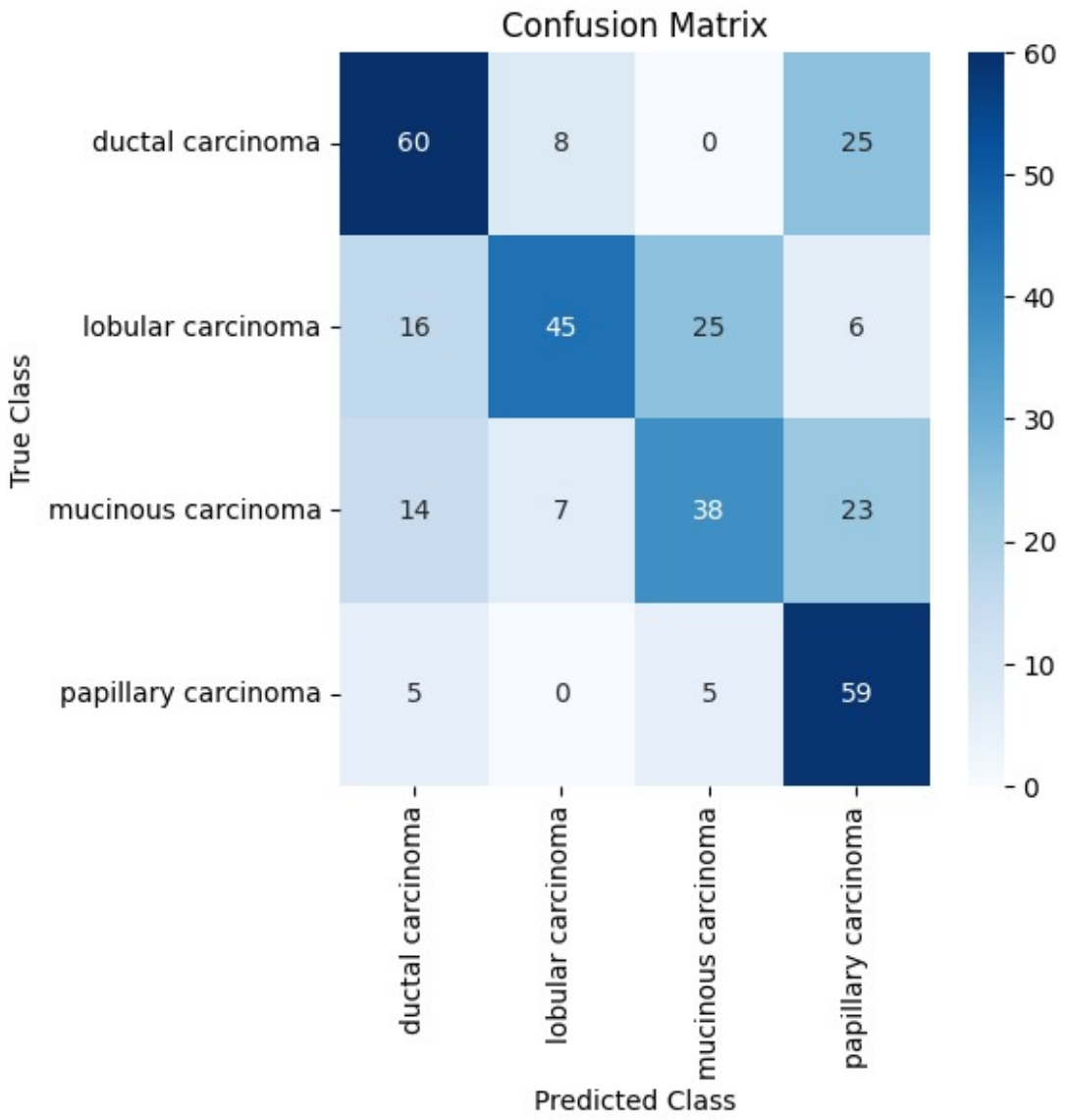



```
# Evaluate the model on the validation/test dataset
val_loss, val_accuracy = custom_model.evaluate(validation_ds)
print(f'Validation Loss: {val_loss}')
print(f'Validation Accuracy: {val_accuracy}')
```

```
# Predict the classes on the validation/test dataset
Y_pred = custom_model.predict(validation_ds)
y_pred = Y_pred.argmax(axis=1)  # Get the index of the max probability
(predicted class)
```

```
12/12 ━━━━━━━━━━━━━━━━━━━━ 2s 205ms/step - accuracy: 0.6285 - loss:
0.9438
Validation Loss: 1.0015554428100586
Validation Accuracy: 0.601190447807312
12/12 ━━━━━━━━━━━━━━━━━━━━ 3s 207ms/step
```

```
#Let's now do a confusion matrix for each model to see how well they
perform

#1) OUR MODEL (CNN)
# Check if y_true needs to be converted
if y_true.ndim == 1:
    y_true_classes = y_true  # If it's already a vector of class
labels
else:
    y_true_classes = np.argmax(y_true, axis=1)  # If it's one-hot
encoded

# y_pred_classes is already computed correctly
y_pred_classes = np.argmax(Y_pred, axis=1)  # Predicted class labels

# Compute the confusion matrix
conf_matrix = confusion_matrix(y_true_classes, y_pred_classes)

# Plot the confusion matrix
class_names = ['ductal carcinoma', 'lobular carcinoma', 'mucinous
carcinoma', 'papillary carcinoma']

plt.figure(figsize=(5, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Class')
plt.ylabel('True Class')
plt.show()
```




```
from keras.utils import to_categorical
from sklearn.metrics import roc_auc_score

# Step 1: Get predicted probabilities from the model
y_pred_prob = custom_model.predict(X_test)  # Predicted probabilities
for each class

# Step 2: Convert y_test to one-hot encoding (if not already done)
y_test_one_hot = to_categorical(y_test, num_classes=4)

# Step 3: Calculate the ROC AUC score for multi-class classification
roc_auc = roc_auc_score(y_test_one_hot, y_pred_prob,
multi_class='ovr')
print(f"ROC AUC: {roc_auc}")
```

```
11/11 ━━━━━━━━━━━━━━━━━━━━ 3s 260ms/step
ROC AUC: 0.8483411566067026
```

```
# Generate the Classification Report
#1) For CNN
report = classification_report(y_true_classes, y_pred_classes,
target_names=class_names)
print('Classification Report: CNN')
print(report)
```

```
Classification Report: CNN
                     precision    recall  f1-score   support

   ductal carcinoma       0.63      0.65      0.64        93
  lobular carcinoma       0.75      0.49      0.59        92
 mucinous carcinoma       0.56      0.46      0.51        82
papillary carcinoma       0.52      0.86      0.65        69

           accuracy                           0.60       336
          macro avg       0.62      0.61      0.60       336
       weighted avg       0.62      0.60      0.60       336
```

```
# Load the ResNet50 model with pre-trained ImageNet weights, excluding
the top layer
ResNet_base_model = ResNet50(weights='imagenet', include_top=False,
input_shape=(224, 224, 3))

# Freeze the layers of the base model
for layer in ResNet_base_model.layers:
    layer.trainable = False

# Add custom layers on top of the base model
x = ResNet_base_model.output

#Global Average Pooling Layer
x = GlobalAveragePooling2D()(x)

#Fully Connected Layer
x = Dense(1024, activation='relu')(x)

#Output Layer
predictions = Dense(4, activation='softmax')(x)

# Create the final model
ResNetmodel = Model(inputs=ResNet_base_model.input,
outputs=predictions)

#Summary of the model
ResNetmodel.summary()
```

```
Model: "functional_3"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━┓
┃ Layer (type)              ┃ Output Shape              ┃         Param # ┃ Connected to              ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━┩
│ input_layer_2             │ (None, 224, 224,          │               0 │ -                         │
│ (InputLayer)              │ 3)                        │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
│ conv1_pad                 │ (None, 230, 230,          │               0 │ input_layer_2[0][…        │
│ (ZeroPadding2D)           │ 3)                        │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
│ conv1_conv (Conv2D)       │ (None, 112, 112,          │           9,472 │ conv1_pad[0][0]           │
│                           │ 64)                       │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
│ conv1_bn                  │ (None, 112, 112,          │             256 │ conv1_conv[0][0]          │
│ (BatchNormalizatio…       │ 64)                       │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
│ conv1_relu                │ (None, 112, 112,          │               0 │ conv1_bn[0][0]            │
│ (Activation)              │ 64)                       │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
│ pool1_pad                 │ (None, 114, 114,          │               0 │ conv1_relu[0][0]          │
│ (ZeroPadding2D)           │ 64)                       │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
│ pool1_pool                │ (None, 56, 56,            │               0 │ pool1_pad[0][0]           │
│ (MaxPooling2D)            │ 64)                       │                 │                           │
├───────────────────────────┼───────────────────────────┼─────────────────┼───────────────────────────┤
```

```
| conv2_block1_1_conv | (None, 56, 56,     |      4,160 | pool1_pool[0]
[0]  |
| (Conv2D)            | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_1_bn   | (None, 56, 56,     |        256 |
conv2_block1_1_c… |
| (BatchNormalizatio… | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_1_relu | (None, 56, 56,     |          0 |
conv2_block1_1_b… |
| (Activation)        | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_2_conv | (None, 56, 56,     |     36,928 |
conv2_block1_1_r… |
| (Conv2D)            | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_2_bn   | (None, 56, 56,     |        256 |
conv2_block1_2_c… |
| (BatchNormalizatio… | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_2_relu | (None, 56, 56,     |          0 |
conv2_block1_2_b… |
| (Activation)        | 64)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_0_conv | (None, 56, 56,     |     16,640 | pool1_pool[0]
[0]  |
| (Conv2D)            | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_3_conv | (None, 56, 56,     |     16,640 |
conv2_block1_2_r… |
| (Conv2D)            | 256)               |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| conv2_block1_0_bn   | (None, 56, 56,     |      1,024 |
```

```
conv2_block1_0_c… │
│ (BatchNormalizatio… │ 256)               │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block1_3_bn   │ (None, 56, 56,     │       1,024 │
conv2_block1_3_c… │
│ (BatchNormalizatio… │ 256)               │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block1_add    │ (None, 56, 56,     │           0 │
conv2_block1_0_b… │
│ (Add)               │ 256)               │             │
conv2_block1_3_b… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block1_out    │ (None, 56, 56,     │           0 │
conv2_block1_add… │
│ (Activation)        │ 256)               │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_1_conv │ (None, 56, 56,     │      16,448 │
conv2_block1_out… │
│ (Conv2D)            │ 64)                │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_1_bn   │ (None, 56, 56,     │         256 │
conv2_block2_1_c… │
│ (BatchNormalizatio… │ 64)                │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_1_relu │ (None, 56, 56,     │           0 │
conv2_block2_1_b… │
│ (Activation)        │ 64)                │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_2_conv │ (None, 56, 56,     │      36,928 │
conv2_block2_1_r… │
│ (Conv2D)            │ 64)                │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv2_block2_2_bn   │ (None, 56, 56,     │         256 │
conv2_block2_2_c… │
```

(BatchNormalizatio…	64)		
conv2_block2_2_relu (Activation)	(None, 56, 56, 64)	0	conv2_block2_2_b…
conv2_block2_3_conv (Conv2D)	(None, 56, 56, 256)	16,640	conv2_block2_2_r…
conv2_block2_3_bn (BatchNormalizatio…	(None, 56, 56, 256)	1,024	conv2_block2_3_c…
conv2_block2_add (Add)	(None, 56, 56, 256)	0	conv2_block1_out… conv2_block2_3_b…
conv2_block2_out (Activation)	(None, 56, 56, 256)	0	conv2_block2_add…
conv2_block3_1_conv (Conv2D)	(None, 56, 56, 64)	16,448	conv2_block2_out…
conv2_block3_1_bn (BatchNormalizatio…	(None, 56, 56, 64)	256	conv2_block3_1_c…
conv2_block3_1_relu (Activation)	(None, 56, 56, 64)	0	conv2_block3_1_b…

```
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_2_conv  │ (None, 56, 56,       │      36,928 │
conv2_block3_1_r… │
│ (Conv2D)             │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_2_bn    │ (None, 56, 56,       │         256 │
conv2_block3_2_c… │
│ (BatchNormalizatio… │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_2_relu  │ (None, 56, 56,       │           0 │
conv2_block3_2_b… │
│ (Activation)         │ 64)                  │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_3_conv  │ (None, 56, 56,       │      16,640 │
conv2_block3_2_r… │
│ (Conv2D)             │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_3_bn    │ (None, 56, 56,       │       1,024 │
conv2_block3_3_c… │
│ (BatchNormalizatio… │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_add     │ (None, 56, 56,       │           0 │
conv2_block2_out… │
│ (Add)                │ 256)                 │             │
conv2_block3_3_b… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv2_block3_out     │ (None, 56, 56,       │           0 │
conv2_block3_add… │
│ (Activation)         │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv3_block1_1_conv  │ (None, 28, 28,       │      32,896 │
conv2_block3_out… │
│ (Conv2D)             │ 128)                 │             │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_1_bn    │ (None, 28, 28,       │          512 │ conv3_block1_1_c… │
│ (BatchNormalizatio… │ 128)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_1_relu  │ (None, 28, 28,       │            0 │ conv3_block1_1_b… │
│ (Activation)         │ 128)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_2_conv  │ (None, 28, 28,       │      147,584 │ conv3_block1_1_r… │
│ (Conv2D)             │ 128)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_2_bn    │ (None, 28, 28,       │          512 │ conv3_block1_2_c… │
│ (BatchNormalizatio… │ 128)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_2_relu  │ (None, 28, 28,       │            0 │ conv3_block1_2_b… │
│ (Activation)         │ 128)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_0_conv  │ (None, 28, 28,       │      131,584 │ conv2_block3_out… │
│ (Conv2D)             │ 512)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_3_conv  │ (None, 28, 28,       │       66,048 │ conv3_block1_2_r… │
│ (Conv2D)             │ 512)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ conv3_block1_0_bn    │ (None, 28, 28,       │        2,048 │ conv3_block1_0_c… │
│ (BatchNormalizatio… │ 512)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
```

```
──────┤
│ conv3_block1_3_bn   │ (None, 28, 28,     │       2,048 │
conv3_block1_3_c… │
│ (BatchNormalizatio… │ 512)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block1_add    │ (None, 28, 28,     │           0 │
conv3_block1_0_b… │
│ (Add)               │ 512)               │             │
conv3_block1_3_b… │
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block1_out    │ (None, 28, 28,     │           0 │
conv3_block1_add… │
│ (Activation)        │ 512)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block2_1_conv │ (None, 28, 28,     │     65,664 │
conv3_block1_out… │
│ (Conv2D)            │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block2_1_bn   │ (None, 28, 28,     │         512 │
conv3_block2_1_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block2_1_relu │ (None, 28, 28,     │           0 │
conv3_block2_1_b… │
│ (Activation)        │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block2_2_conv │ (None, 28, 28,     │    147,584 │
conv3_block2_1_r… │
│ (Conv2D)            │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
│ conv3_block2_2_bn   │ (None, 28, 28,     │         512 │
conv3_block2_2_c… │
│ (BatchNormalizatio… │ 128)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────
──────┤
```

```
| conv3_block2_2_relu | (None, 28, 28,    |           0 |
conv3_block2_2_b… |
| (Activation)        | 128)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block2_3_conv | (None, 28, 28,    |     66,048 |
conv3_block2_2_r… |
| (Conv2D)            | 512)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block2_3_bn   | (None, 28, 28,    |      2,048 |
conv3_block2_3_c… |
| (BatchNormalizatio… | 512)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block2_add    | (None, 28, 28,    |           0 |
conv3_block1_out… |
| (Add)               | 512)              |             |
conv3_block2_3_b… |
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block2_out    | (None, 28, 28,    |           0 |
conv3_block2_add… |
| (Activation)        | 512)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_1_conv | (None, 28, 28,    |     65,664 |
conv3_block2_out… |
| (Conv2D)            | 128)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_1_bn   | (None, 28, 28,    |         512 |
conv3_block3_1_c… |
| (BatchNormalizatio… | 128)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_1_relu | (None, 28, 28,    |           0 |
conv3_block3_1_b… |
| (Activation)        | 128)              |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_2_conv | (None, 28, 28,    |    147,584 |
```

```
conv3_block3_1_r… |
| (Conv2D)              | 128)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_2_bn    | (None, 28, 28,      |         512 |
conv3_block3_2_c… |
| (BatchNormalizatio… | 128)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_2_relu | (None, 28, 28,      |           0 |
conv3_block3_2_b… |
| (Activation)          | 128)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_3_conv | (None, 28, 28,      |     66,048 |
conv3_block3_2_r… |
| (Conv2D)              | 512)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_3_bn    | (None, 28, 28,      |      2,048 |
conv3_block3_3_c… |
| (BatchNormalizatio… | 512)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_add     | (None, 28, 28,      |           0 |
conv3_block2_out… |
| (Add)                 | 512)                |             |
conv3_block3_3_b… |
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block3_out     | (None, 28, 28,      |           0 |
conv3_block3_add… |
| (Activation)          | 512)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block4_1_conv | (None, 28, 28,      |     65,664 |
conv3_block3_out… |
| (Conv2D)              | 128)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
| conv3_block4_1_bn    | (None, 28, 28,      |         512 |
conv3_block4_1_c… |
```

```
| (BatchNormalizatio… | 128)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_1_relu | (None, 28, 28,    |           0 |
conv3_block4_1_b… |
| (Activation)        | 128)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_2_conv | (None, 28, 28,    |     147,584 |
conv3_block4_1_r… |
| (Conv2D)            | 128)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_2_bn   | (None, 28, 28,    |         512 |
conv3_block4_2_c… |
| (BatchNormalizatio… | 128)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_2_relu | (None, 28, 28,    |           0 |
conv3_block4_2_b… |
| (Activation)        | 128)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_3_conv | (None, 28, 28,    |      66,048 |
conv3_block4_2_r… |
| (Conv2D)            | 512)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_3_bn   | (None, 28, 28,    |       2,048 |
conv3_block4_3_c… |
| (BatchNormalizatio… | 512)              |             |
|                     |                   |             |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_add    | (None, 28, 28,    |           0 |
conv3_block3_out… |
| (Add)               | 512)              |             |
conv3_block4_3_b… |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| conv3_block4_out    | (None, 28, 28,    |           0 |
conv3_block4_add… |
| (Activation)        | 512)              |             |
```

```
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_1_conv │ (None, 14, 14,     │     131,328 │
conv3_block4_out… │
│ (Conv2D)            │ 256)               │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_1_bn   │ (None, 14, 14,     │       1,024 │
conv4_block1_1_c… │
│ (BatchNormalizatio… │ 256)               │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_1_relu │ (None, 14, 14,     │           0 │
conv4_block1_1_b… │
│ (Activation)        │ 256)               │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_2_conv │ (None, 14, 14,     │     590,080 │
conv4_block1_1_r… │
│ (Conv2D)            │ 256)               │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_2_bn   │ (None, 14, 14,     │       1,024 │
conv4_block1_2_c… │
│ (BatchNormalizatio… │ 256)               │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_2_relu │ (None, 14, 14,     │           0 │
conv4_block1_2_b… │
│ (Activation)        │ 256)               │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_0_conv │ (None, 14, 14,     │     525,312 │
conv3_block4_out… │
│ (Conv2D)            │ 1024)              │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
───────┤
│ conv4_block1_3_conv │ (None, 14, 14,     │     263,168 │
conv4_block1_2_r… │
│ (Conv2D)            │ 1024)              │             │
│
├─────────────────────────┼────────────────────┼─────────────┼───────────
```

```
──────┤
│ conv4_block1_0_bn   │ (None, 14, 14,     │       4,096 │
conv4_block1_0_c… │
│ (BatchNormalizatio… │ 1024)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block1_3_bn   │ (None, 14, 14,     │       4,096 │
conv4_block1_3_c… │
│ (BatchNormalizatio… │ 1024)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block1_add    │ (None, 14, 14,     │           0 │
conv4_block1_0_b… │
│ (Add)               │ 1024)              │             │
conv4_block1_3_b… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block1_out    │ (None, 14, 14,     │           0 │
conv4_block1_add… │
│ (Activation)        │ 1024)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block2_1_conv │ (None, 14, 14,     │     262,400 │
conv4_block1_out… │
│ (Conv2D)            │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block2_1_bn   │ (None, 14, 14,     │       1,024 │
conv4_block2_1_c… │
│ (BatchNormalizatio… │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block2_1_relu │ (None, 14, 14,     │           0 │
conv4_block2_1_b… │
│ (Activation)        │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block2_2_conv │ (None, 14, 14,     │     590,080 │
conv4_block2_1_r… │
│ (Conv2D)            │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| conv4_block2_2_bn   | (None, 14, 14,     |       1,024 |
conv4_block2_2_c… |
| (BatchNormalizatio… | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block2_2_relu | (None, 14, 14,     |           0 |
conv4_block2_2_b… |
| (Activation)        | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block2_3_conv | (None, 14, 14,     |     263,168 |
conv4_block2_2_r… |
| (Conv2D)            | 1024)              |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block2_3_bn   | (None, 14, 14,     |       4,096 |
conv4_block2_3_c… |
| (BatchNormalizatio… | 1024)              |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block2_add    | (None, 14, 14,     |           0 |
conv4_block1_out… |
| (Add)               | 1024)              |             |
conv4_block2_3_b… |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block2_out    | (None, 14, 14,     |           0 |
conv4_block2_add… |
| (Activation)        | 1024)              |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block3_1_conv | (None, 14, 14,     |     262,400 |
conv4_block2_out… |
| (Conv2D)            | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block3_1_bn   | (None, 14, 14,     |       1,024 |
conv4_block3_1_c… |
| (BatchNormalizatio… | 256)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv4_block3_1_relu | (None, 14, 14,     |           0 |
```

```
conv4_block3_1_b… │
│ (Activation)         │ 256)                │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_2_conv  │ (None, 14, 14,      │    590,080 │
conv4_block3_1_r… │
│ (Conv2D)             │ 256)                │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_2_bn    │ (None, 14, 14,      │      1,024 │
conv4_block3_2_c… │
│ (BatchNormalizatio… │ 256)                │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_2_relu  │ (None, 14, 14,      │          0 │
conv4_block3_2_b… │
│ (Activation)         │ 256)                │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_3_conv  │ (None, 14, 14,      │    263,168 │
conv4_block3_2_r… │
│ (Conv2D)             │ 1024)               │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_3_bn    │ (None, 14, 14,      │      4,096 │
conv4_block3_3_c… │
│ (BatchNormalizatio… │ 1024)               │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_add     │ (None, 14, 14,      │          0 │
conv4_block2_out… │
│ (Add)                │ 1024)               │             │
conv4_block3_3_b… │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block3_out     │ (None, 14, 14,      │          0 │
conv4_block3_add… │
│ (Activation)         │ 1024)               │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block4_1_conv  │ (None, 14, 14,      │    262,400 │
conv4_block3_out… │
```

```
| (Conv2D)              | 256)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_1_bn     | (None, 14, 14,       |       1,024 |
conv4_block4_1_c… |
| (BatchNormalizatio… | 256)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_1_relu | (None, 14, 14,       |           0 |
conv4_block4_1_b… |
| (Activation)          | 256)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_2_conv | (None, 14, 14,       |     590,080 |
conv4_block4_1_r… |
| (Conv2D)              | 256)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_2_bn     | (None, 14, 14,       |       1,024 |
conv4_block4_2_c… |
| (BatchNormalizatio… | 256)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_2_relu | (None, 14, 14,       |           0 |
conv4_block4_2_b… |
| (Activation)          | 256)                 |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_3_conv | (None, 14, 14,       |     263,168 |
conv4_block4_2_r… |
| (Conv2D)              | 1024)                |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_3_bn     | (None, 14, 14,       |       4,096 |
conv4_block4_3_c… |
| (BatchNormalizatio… | 1024)                |             |
|                       |                      |             |
├───────────────────────┼──────────────────────┼─────────────┼──────────────
──────┤
| conv4_block4_add      | (None, 14, 14,       |           0 |
conv4_block3_out… |
| (Add)                 | 1024)                |             |
|                       |                      |             |
```

```
conv4_block4_3_b… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block4_out     │ (None, 14, 14,       │            0 │
conv4_block4_add… │
│ (Activation)         │ 1024)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_1_conv  │ (None, 14, 14,       │      262,400 │
conv4_block4_out… │
│ (Conv2D)             │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_1_bn    │ (None, 14, 14,       │        1,024 │
conv4_block5_1_c… │
│ (BatchNormalizatio… │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_1_relu  │ (None, 14, 14,       │            0 │
conv4_block5_1_b… │
│ (Activation)         │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_2_conv  │ (None, 14, 14,       │      590,080 │
conv4_block5_1_r… │
│ (Conv2D)             │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_2_bn    │ (None, 14, 14,       │        1,024 │
conv4_block5_2_c… │
│ (BatchNormalizatio… │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_2_relu  │ (None, 14, 14,       │            0 │
conv4_block5_2_b… │
│ (Activation)         │ 256)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_3_conv  │ (None, 14, 14,       │      263,168 │
conv4_block5_2_r… │
│ (Conv2D)             │ 1024)                │              │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_3_bn    │ (None, 14, 14,       │       4,096 │
conv4_block5_3_c… │
│ (BatchNormalizatio… │ 1024)                │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_add     │ (None, 14, 14,       │           0 │
conv4_block4_out… │
│ (Add)                │ 1024)                │             │
conv4_block5_3_b… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block5_out     │ (None, 14, 14,       │           0 │
conv4_block5_add… │
│ (Activation)         │ 1024)                │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block6_1_conv  │ (None, 14, 14,       │     262,400 │
conv4_block5_out… │
│ (Conv2D)             │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block6_1_bn    │ (None, 14, 14,       │       1,024 │
conv4_block6_1_c… │
│ (BatchNormalizatio… │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block6_1_relu  │ (None, 14, 14,       │           0 │
conv4_block6_1_b… │
│ (Activation)         │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block6_2_conv  │ (None, 14, 14,       │     590,080 │
conv4_block6_1_r… │
│ (Conv2D)             │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ conv4_block6_2_bn    │ (None, 14, 14,       │       1,024 │
conv4_block6_2_c… │
│ (BatchNormalizatio… │ 256)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
```

```
──────┤
│ conv4_block6_2_relu │ (None, 14, 14,     │           0 │
conv4_block6_2_b… │
│ (Activation)        │ 256)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block6_3_conv │ (None, 14, 14,     │     263,168 │
conv4_block6_2_r… │
│ (Conv2D)            │ 1024)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block6_3_bn   │ (None, 14, 14,     │       4,096 │
conv4_block6_3_c… │
│ (BatchNormalizatio… │ 1024)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block6_add    │ (None, 14, 14,     │           0 │
conv4_block5_out… │
│ (Add)               │ 1024)              │             │
conv4_block6_3_b… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv4_block6_out    │ (None, 14, 14,     │           0 │
conv4_block6_add… │
│ (Activation)        │ 1024)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block1_1_conv │ (None, 7, 7, 512)  │     524,800 │
conv4_block6_out… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block1_1_bn   │ (None, 7, 7, 512)  │       2,048 │
conv5_block1_1_c… │
│ (BatchNormalizatio… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block1_1_relu │ (None, 7, 7, 512)  │           0 │
conv5_block1_1_b… │
│ (Activation)        │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| conv5_block1_2_conv | (None, 7, 7, 512) |  2,359,808 |
conv5_block1_1_r… |
| (Conv2D)            |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_2_bn   | (None, 7, 7, 512) |      2,048 |
conv5_block1_2_c… |
| (BatchNormalizatio… |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_2_relu | (None, 7, 7, 512) |          0 |
conv5_block1_2_b… |
| (Activation)        |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_0_conv | (None, 7, 7,      |  2,099,200 |
conv4_block6_out… |
| (Conv2D)            | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_3_conv | (None, 7, 7,      |  1,050,624 |
conv5_block1_2_r… |
| (Conv2D)            | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_0_bn   | (None, 7, 7,      |      8,192 |
conv5_block1_0_c… |
| (BatchNormalizatio… | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_3_bn   | (None, 7, 7,      |      8,192 |
conv5_block1_3_c… |
| (BatchNormalizatio… | 2048)             |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_add    | (None, 7, 7,      |          0 |
conv5_block1_0_b… |
| (Add)               | 2048)             |            |
conv5_block1_3_b… |
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| conv5_block1_out    | (None, 7, 7,      |          0 |
```

```
conv5_block1_add… │
│ (Activation)         │ 2048)               │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_1_conv  │ (None, 7, 7, 512) │  1,049,088 │
conv5_block1_out… │
│ (Conv2D)             │                     │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_1_bn    │ (None, 7, 7, 512) │      2,048 │
conv5_block2_1_c… │
│ (BatchNormalizatio… │                     │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_1_relu  │ (None, 7, 7, 512) │          0 │
conv5_block2_1_b… │
│ (Activation)         │                     │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_2_conv  │ (None, 7, 7, 512) │  2,359,808 │
conv5_block2_1_r… │
│ (Conv2D)             │                     │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_2_bn    │ (None, 7, 7, 512) │      2,048 │
conv5_block2_2_c… │
│ (BatchNormalizatio… │                     │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_2_relu  │ (None, 7, 7, 512) │          0 │
conv5_block2_2_b… │
│ (Activation)         │                     │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_3_conv  │ (None, 7, 7,       │  1,050,624 │
conv5_block2_2_r… │
│ (Conv2D)             │ 2048)               │             │
│
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ conv5_block2_3_bn    │ (None, 7, 7,       │      8,192 │
conv5_block2_3_c… │
```

```
| (BatchNormalizatio… | 2048)              |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block2_add    | (None, 7, 7,       |           0 |
conv5_block1_out… |
| (Add)               | 2048)              |             |
conv5_block2_3_b… |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block2_out    | (None, 7, 7,       |           0 |
conv5_block2_add… |
| (Activation)        | 2048)              |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block3_1_conv | (None, 7, 7, 512) |  1,049,088 |
conv5_block2_out… |
| (Conv2D)            |                    |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block3_1_bn   | (None, 7, 7, 512) |       2,048 |
conv5_block3_1_c… |
| (BatchNormalizatio… |                    |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block3_1_relu | (None, 7, 7, 512) |           0 |
conv5_block3_1_b… |
| (Activation)        |                    |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block3_2_conv | (None, 7, 7, 512) |  2,359,808 |
conv5_block3_1_r… |
| (Conv2D)            |                    |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block3_2_bn   | (None, 7, 7, 512) |       2,048 |
conv5_block3_2_c… |
| (BatchNormalizatio… |                    |             |
|                     |                    |             |
├─────────────────────┼────────────────────┼─────────────┼──────────────
──────┤
| conv5_block3_2_relu | (None, 7, 7, 512) |           0 |
conv5_block3_2_b… |
| (Activation)        |                    |             |
```

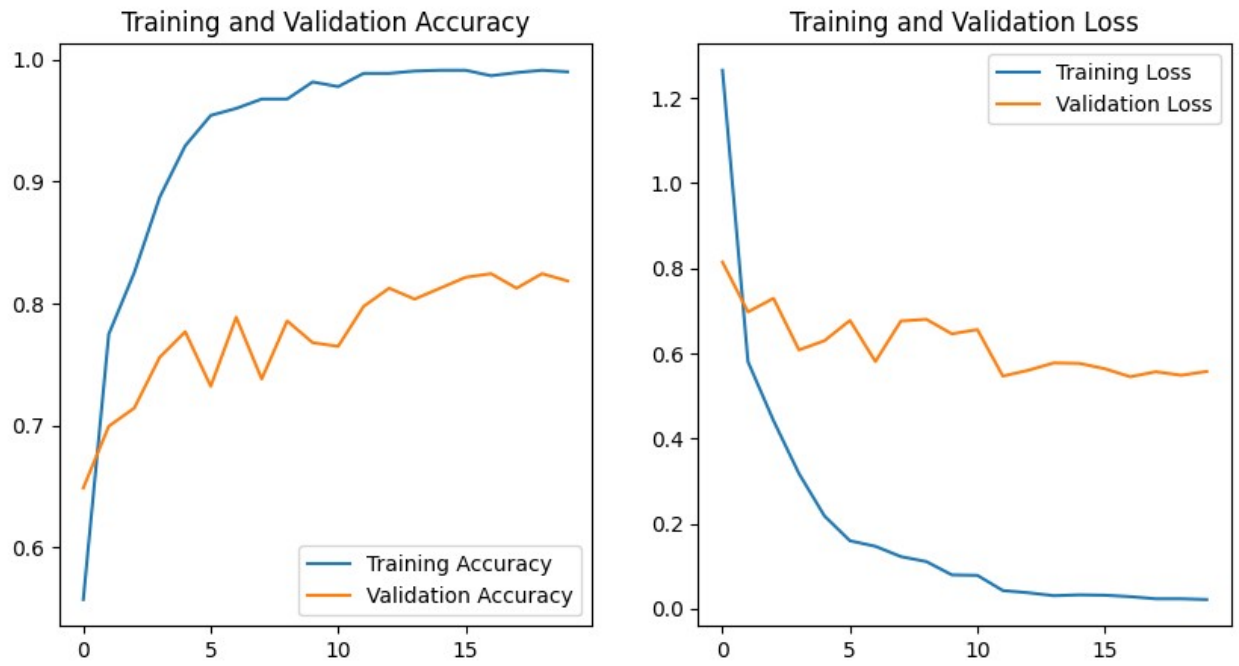
```
│
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ conv5_block3_3_conv │ (None, 7, 7, │ 1,050,624 │
conv5_block3_2_r… │
│ (Conv2D) │ 2048) │ │
│
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ conv5_block3_3_bn │ (None, 7, 7, │ 8,192 │
conv5_block3_3_c… │
│ (BatchNormalizatio… │ 2048) │ │
│
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ conv5_block3_add │ (None, 7, 7, │ 0 │
conv5_block2_out… │
│ (Add) │ 2048) │ │
conv5_block3_3_b… │
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ conv5_block3_out │ (None, 7, 7, │ 0 │
conv5_block3_add… │
│ (Activation) │ 2048) │ │
│
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ global_average_poo… │ (None, 2048) │ 0 │
conv5_block3_out… │
│ (GlobalAveragePool… │ │ │
│
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ dense_3 (Dense) │ (None, 1024) │ 2,098,176 │
global_average_p… │
├───────────────────────────┼─────────────────────────┼────────────────┼───────────────────
────────┤
│ dense_4 (Dense) │ (None, 4) │ 4,100 │ dense_3[0][0]
│
└───────────────────────────┴─────────────────────────┴────────────────┴───────────────────
────────┘

 Total params: 25,689,988 (98.00 MB)

 Trainable params: 2,102,276 (8.02 MB)

 Non-trainable params: 23,587,712 (89.98 MB)

# Compile the model
ResNetmodel.compile(optimizer=Adam(learning_rate=0.001),
```

```
              loss='sparse_categorical_crossentropy',  # Suitable for
multi-class classification
              metrics=["accuracy"])

#Fit the model
epochs=20
history = ResNetmodel.fit(train_ds, validation_data=validation_ds,
epochs=epochs, callbacks = [early_stopping, reduce_lr])
```

```
Epoch 1/20

56/56 ━━━━━━━━━━━━━━━━━━━━ 182s 3s/step - accuracy: 0.4627 - loss:
1.7644 - val_accuracy: 0.6488 - val_loss: 0.8145 - learning_rate:
0.0010
Epoch 2/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 171s 3s/step - accuracy: 0.7675 - loss:
0.5897 - val_accuracy: 0.6994 - val_loss: 0.6973 - learning_rate:
0.0010
Epoch 3/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 186s 3s/step - accuracy: 0.8497 - loss:
0.4001 - val_accuracy: 0.7143 - val_loss: 0.7293 - learning_rate:
0.0010
Epoch 4/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 195s 4s/step - accuracy: 0.9002 - loss:
0.3087 - val_accuracy: 0.7560 - val_loss: 0.6080 - learning_rate:
0.0010
Epoch 5/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 183s 3s/step - accuracy: 0.9336 - loss:
0.2123 - val_accuracy: 0.7768 - val_loss: 0.6297 - learning_rate:
0.0010
Epoch 6/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 186s 3s/step - accuracy: 0.9484 - loss:
0.1711 - val_accuracy: 0.7321 - val_loss: 0.6773 - learning_rate:
0.0010
Epoch 7/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 184s 3s/step - accuracy: 0.9646 - loss:
0.1349 - val_accuracy: 0.7887 - val_loss: 0.5808 - learning_rate:
0.0010
Epoch 8/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 189s 3s/step - accuracy: 0.9759 - loss:
0.1054 - val_accuracy: 0.7381 - val_loss: 0.6760 - learning_rate:
0.0010
Epoch 9/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 187s 3s/step - accuracy: 0.9736 - loss:
0.0972 - val_accuracy: 0.7857 - val_loss: 0.6799 - learning_rate:
0.0010
Epoch 10/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 188s 3s/step - accuracy: 0.9851 - loss:
0.0735 - val_accuracy: 0.7679 - val_loss: 0.6459 - learning_rate:
0.0010
```

```
Epoch 11/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9797 - loss:
0.0739
Epoch 11: ReduceLROnPlateau reducing learning rate to
0.0005000000237487257.
56/56 ━━━━━━━━━━━━━━━━━━━━ 190s 3s/step - accuracy: 0.9796 - loss:
0.0740 - val_accuracy: 0.7649 - val_loss: 0.6559 - learning_rate:
0.0010
Epoch 12/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 208s 4s/step - accuracy: 0.9916 - loss:
0.0403 - val_accuracy: 0.7976 - val_loss: 0.5466 - learning_rate:
5.0000e-04
Epoch 13/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 188s 3s/step - accuracy: 0.9928 - loss:
0.0306 - val_accuracy: 0.8125 - val_loss: 0.5603 - learning_rate:
5.0000e-04
Epoch 14/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 185s 3s/step - accuracy: 0.9946 - loss:
0.0243 - val_accuracy: 0.8036 - val_loss: 0.5775 - learning_rate:
5.0000e-04
Epoch 15/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 185s 3s/step - accuracy: 0.9933 - loss:
0.0296 - val_accuracy: 0.8125 - val_loss: 0.5764 - learning_rate:
5.0000e-04
Epoch 16/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 0s 3s/step - accuracy: 0.9898 - loss:
0.0350
Epoch 16: ReduceLROnPlateau reducing learning rate to
0.0002500000118743628.
56/56 ━━━━━━━━━━━━━━━━━━━━ 197s 4s/step - accuracy: 0.9899 - loss:
0.0350 - val_accuracy: 0.8214 - val_loss: 0.5639 - learning_rate:
5.0000e-04
Epoch 17/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 197s 4s/step - accuracy: 0.9837 - loss:
0.0317 - val_accuracy: 0.8244 - val_loss: 0.5451 - learning_rate:
2.5000e-04
Epoch 18/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 183s 3s/step - accuracy: 0.9904 - loss:
0.0222 - val_accuracy: 0.8125 - val_loss: 0.5568 - learning_rate:
2.5000e-04
Epoch 19/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 182s 3s/step - accuracy: 0.9919 - loss:
0.0217 - val_accuracy: 0.8244 - val_loss: 0.5486 - learning_rate:
2.5000e-04
Epoch 20/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 188s 3s/step - accuracy: 0.9869 - loss:
0.0270 - val_accuracy: 0.8185 - val_loss: 0.5575 - learning_rate:
2.5000e-04
Restoring model weights from the end of the best epoch: 17.
```

```
ResNet_acc = history.history['accuracy']
ResNet_val_acc = history.history['val_accuracy']

ResNet_loss = history.history['loss']
ResNet_val_loss = history.history['val_loss']

epochs_range = range(len(history.history['accuracy']))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, ResNet_acc, label='Training Accuracy')
plt.plot(epochs_range, ResNet_val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, ResNet_loss, label='Training Loss')
plt.plot(epochs_range, ResNet_val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```

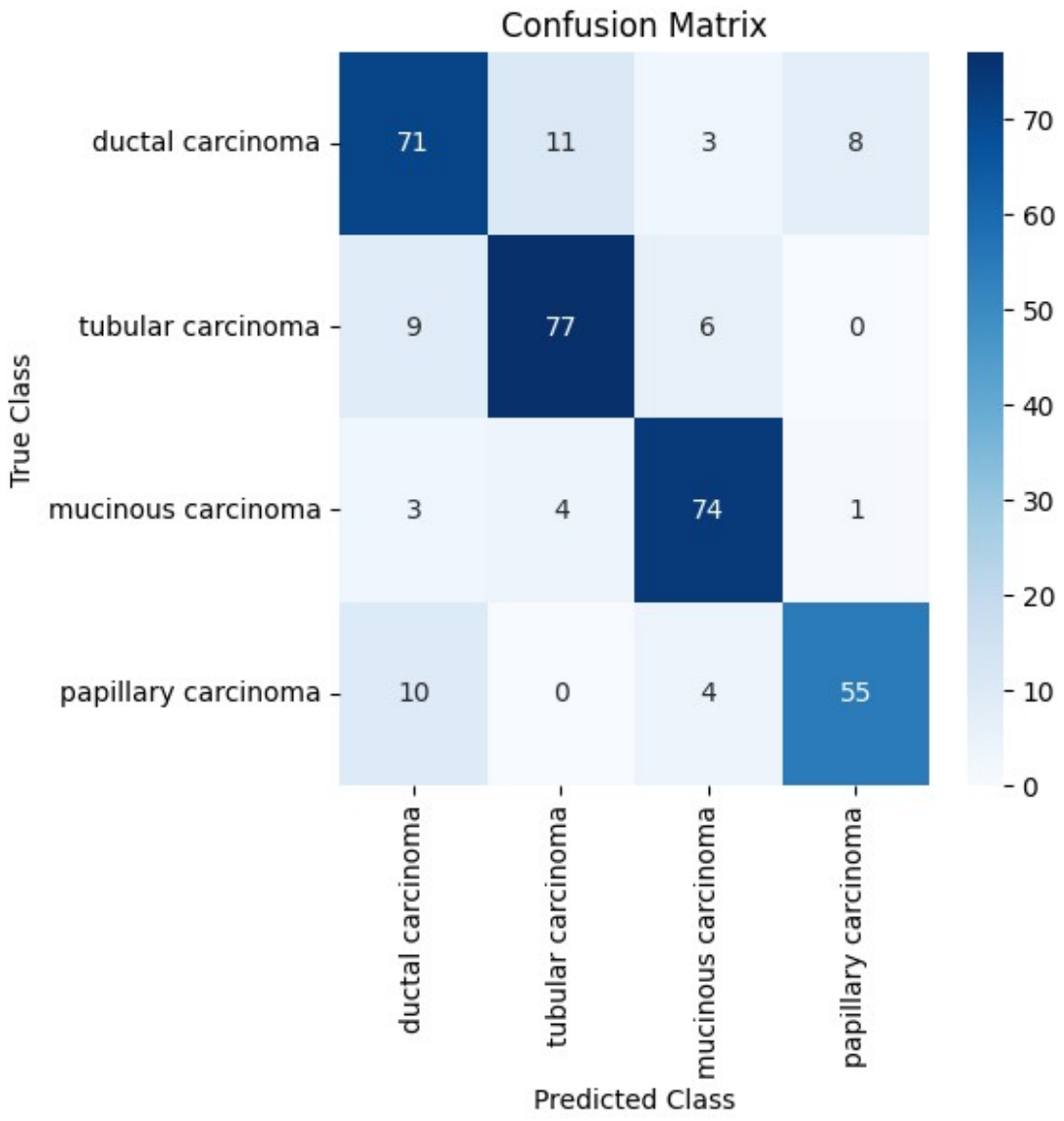



```
# Evaluate the model on the validation/test dataset
ResNet_val_loss, ResNet_val_accuracy =
ResNetmodel.evaluate(validation_ds)
print(f'ResNet Validation Loss: {ResNet_val_loss}')
print(f'ResNet Validation Accuracy: {ResNet_val_accuracy}')
```

```
# Predict the classes on the validation/test dataset
ResNet_Y_pred = ResNetmodel.predict(validation_ds)
ResNet_y_pred = ResNet_Y_pred.argmax(axis=1)  # Get the index of the
max probability (predicted class)
```

```
12/12 ━━━━━━━━━━━━━━━━━━━━ 33s 3s/step - accuracy: 0.8448 - loss:
0.4398
ResNet Validation Loss: 0.5450805425643921
ResNet Validation Accuracy: 0.824404776096344
12/12 ━━━━━━━━━━━━━━━━━━━━ 39s 3s/step
```

```
#2) ResNet
#  Check if y_true needs to be converted
if y_true.ndim == 1:
    ResNet_y_true_classes = y_true  # If it's already a vector of
class labels
else:
    ResNet_y_true_classes = np.argmax(y_true, axis=1)  # If it's one-
hot encoded

# y_pred_classes is already computed correctly
ResNet_y_pred_classes = np.argmax(ResNet_Y_pred, axis=1)  # Predicted
class labels

# Compute the confusion matrix
conf_matrix = confusion_matrix(ResNet_y_true_classes,
ResNet_y_pred_classes)

# Plot the confusion matrix
# Plot the confusion matrix
class_names = ['ductal carcinoma', 'tubular carcinoma', 'mucinous
carcinoma', 'papillary carcinoma']

plt.figure(figsize=(5, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Class')
plt.ylabel('True Class')
plt.show()
```




```
#predicted probabilities for each class
y_pred_prob = ResNetmodel.predict(X_test)

# Convert y_test to one-hot encoded
y_test_one_hot = to_categorical(y_test, num_classes=4)

# Calculate the ROC AUC score
roc_auc = roc_auc_score(y_test_one_hot, y_pred_prob,
multi_class='ovr')
print(f"ROC AUC: {roc_auc}")

11/11 ━━━━━━━━━━━━━━━━━━━━ 46s 4s/step
ROC AUC: 0.9418274881121529
```

```
# Classification Report
#2) ResNet
report = classification_report(ResNet_y_true_classes,
ResNet_y_pred_classes, target_names=class_names)
print('Classification Report: ResNet')
print(report)
```

```
Classification Report: ResNet
                     precision    recall  f1-score   support

   ductal carcinoma       0.76      0.76      0.76        93
  tubular carcinoma       0.84      0.84      0.84        92
 mucinous carcinoma       0.85      0.90      0.88        82
papillary carcinoma       0.86      0.80      0.83        69

           accuracy                           0.82       336
          macro avg       0.83      0.82      0.83       336
       weighted avg       0.82      0.82      0.82       336
```

```
# Load the ResNet50 model with pre-trained ImageNet weights, excluding
the top layer
# Load the EfficientNetB0 model with pre-trained ImageNet weights,
excluding the top layer
EfficientNetB0_base_model = EfficientNetB0(weights='imagenet',
include_top=False, input_shape=(224, 224, 3))


# Freeze the layers of the base model
for layer in EfficientNetB0_base_model.layers:
    layer.trainable = False

# Add custom layers on top of the base model
x = EfficientNetB0_base_model.output

#Global Average Pooling Layer
x = GlobalAveragePooling2D()(x)

#Fully Connected Layer
x = Dense(1024, activation='relu')(x)

#Output Layer
predictions = Dense(4, activation='softmax')(x)

# Create the final model
EfficientNetB0model = Model(inputs=EfficientNetB0_base_model.input,
outputs=predictions)

#Summary of the model
EfficientNetB0model.summary()
```

Model: "functional_5"

Layer (type)	Output Shape	Param #	Connected to
input_layer_3 (InputLayer)	(None, 224, 224, 3)	0	-
rescaling_2 (Rescaling)	(None, 224, 224, 3)	0	input_layer_3[0]…
normalization (Normalization)	(None, 224, 224, 3)	7	rescaling_2[0][0]
rescaling_3 (Rescaling)	(None, 224, 224, 3)	0	normalization[0]…
stem_conv_pad (ZeroPadding2D)	(None, 225, 225, 3)	0	rescaling_3[0][0]
stem_conv (Conv2D)	(None, 112, 112, 32)	864	stem_conv_pad[0]…
stem_bn (BatchNormalizatio…	(None, 112, 112, 32)	128	stem_conv[0][0]

```
| stem_activation     | (None, 112, 112,  |          0 | stem_bn[0][0]
|
| (Activation)        | 32)               |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_dwconv      | (None, 112, 112,  |        288 |
stem_activation[… |
| (DepthwiseConv2D)   | 32)               |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_bn          | (None, 112, 112,  |        128 |
block1a_dwconv[0… |
| (BatchNormalizatio… | 32)               |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_activation  | (None, 112, 112,  |          0 | block1a_bn[0]
[0]  |
| (Activation)        | 32)               |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_se_squeeze  | (None, 32)        |          0 |
block1a_activati… |
| (GlobalAveragePool… |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_se_reshape  | (None, 1, 1, 32)  |          0 |
block1a_se_squee… |
| (Reshape)           |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_se_reduce   | (None, 1, 1, 8)   |        264 |
block1a_se_resha… |
| (Conv2D)            |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_se_expand   | (None, 1, 1, 32)  |        288 |
block1a_se_reduc… |
| (Conv2D)            |                   |            |
|
├─────────────────────┼───────────────────┼────────────┼──────────────
──────┤
| block1a_se_excite   | (None, 112, 112,  |          0 |
```

```
block1a_activati… |
| (Multiply)          | 32)                |            |
block1a_se_expan… |
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block1a_project_co… | (None, 112, 112,   |        512 |
block1a_se_excit… |
| (Conv2D)            | 16)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block1a_project_bn  | (None, 112, 112,   |         64 |
block1a_project_… |
| (BatchNormalizatio… | 16)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block2a_expand_conv | (None, 112, 112,   |      1,536 |
block1a_project_… |
| (Conv2D)            | 96)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block2a_expand_bn   | (None, 112, 112,   |        384 |
block2a_expand_c… |
| (BatchNormalizatio… | 96)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block2a_expand_act… | (None, 112, 112,   |          0 |
block2a_expand_b… |
| (Activation)        | 96)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block2a_dwconv_pad  | (None, 113, 113,   |          0 |
block2a_expand_a… |
| (ZeroPadding2D)     | 96)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block2a_dwconv      | (None, 56, 56,     |        864 |
block2a_dwconv_p… |
| (DepthwiseConv2D)   | 96)                |            |
|
├─────────────────────┼────────────────────┼────────────┼──────────────────
──────┤
| block2a_bn          | (None, 56, 56,     |        384 |
block2a_dwconv[0… |
```

```
| (BatchNormalizatio… | 96)                 |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_activation  | (None, 56, 56,      |           0 | block2a_bn[0]
[0]  |
| (Activation)        | 96)                 |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_squeeze  | (None, 96)          |           0 |
block2a_activati… |
| (GlobalAveragePool… |                     |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_reshape  | (None, 1, 1, 96)    |           0 |
block2a_se_squee… |
| (Reshape)           |                     |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_reduce   | (None, 1, 1, 4)     |         388 |
block2a_se_resha… |
| (Conv2D)            |                     |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_expand   | (None, 1, 1, 96)    |         480 |
block2a_se_reduc… |
| (Conv2D)            |                     |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_se_excite   | (None, 56, 56,      |           0 |
block2a_activati… |
| (Multiply)          | 96)                 |             |
block2a_se_expan… |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_project_co… | (None, 56, 56,      |       2,304 |
block2a_se_excit… |
| (Conv2D)            | 24)                 |             |
├─────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block2a_project_bn  | (None, 56, 56,      |          96 |
block2a_project_… |
| (BatchNormalizatio… | 24)                 |             |
```

```
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_expand_conv  │ (None, 56, 56,       │       3,456 │
block2a_project_… │
│ (Conv2D)             │ 144)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_expand_bn    │ (None, 56, 56,       │         576 │
block2b_expand_c… │
│ (BatchNormalizatio…  │ 144)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_expand_act…  │ (None, 56, 56,       │           0 │
block2b_expand_b… │
│ (Activation)         │ 144)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_dwconv       │ (None, 56, 56,       │       1,296 │
block2b_expand_a… │
│ (DepthwiseConv2D)    │ 144)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_bn           │ (None, 56, 56,       │         576 │
block2b_dwconv[0… │
│ (BatchNormalizatio…  │ 144)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_activation   │ (None, 56, 56,       │           0 │ block2b_bn[0]
[0]  │
│ (Activation)         │ 144)                 │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_se_squeeze   │ (None, 144)          │           0 │
block2b_activati… │
│ (GlobalAveragePool…  │                      │             │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block2b_se_reshape   │ (None, 1, 1, 144)    │           0 │
block2b_se_squee… │
│ (Reshape)            │                      │             │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_se_reduce    │ (None, 1, 1, 6)      │          870 │ block2b_se_resha… │
│ (Conv2D)             │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_se_expand    │ (None, 1, 1, 144)    │        1,008 │ block2b_se_reduc… │
│ (Conv2D)             │                      │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_se_excite    │ (None, 56, 56,       │            0 │ block2b_activati… │
│ (Multiply)           │ 144)                 │              │ block2b_se_expan… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_project_co… │ (None, 56, 56,       │        3,456 │ block2b_se_excit… │
│ (Conv2D)             │ 24)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_project_bn   │ (None, 56, 56,       │           96 │ block2b_project_… │
│ (BatchNormalizatio… │ 24)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_drop         │ (None, 56, 56,       │            0 │ block2b_project_… │
│ (Dropout)            │ 24)                  │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block2b_add (Add)    │ (None, 56, 56,       │            0 │ block2b_drop[0][… │
│                      │ 24)                  │              │ block2a_project_… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
│ block3a_expand_conv  │ (None, 56, 56,       │        3,456 │ block2b_add[0][0] │
│ (Conv2D)             │ 144)                 │              │                      │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────────┤
```

```
──────┤
| block3a_expand_bn   | (None, 56, 56,     |         576 |
block3a_expand_c… |
| (BatchNormalizatio… | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_expand_act… | (None, 56, 56,     |           0 |
block3a_expand_b… |
| (Activation)        | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_dwconv_pad  | (None, 59, 59,     |           0 |
block3a_expand_a… |
| (ZeroPadding2D)     | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_dwconv      | (None, 28, 28,     |       3,600 |
block3a_dwconv_p… |
| (DepthwiseConv2D)   | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_bn          | (None, 28, 28,     |         576 |
block3a_dwconv[0… |
| (BatchNormalizatio… | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_activation  | (None, 28, 28,     |           0 | block3a_bn[0]
[0]  |
| (Activation)        | 144)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_se_squeeze  | (None, 144)        |           0 |
block3a_activati… |
| (GlobalAveragePool… |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3a_se_reshape  | (None, 1, 1, 144)  |           0 |
block3a_se_squee… |
| (Reshape)           |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

block3a_se_reduce (Conv2D)	(None, 1, 1, 6)	870	block3a_se_resha…
block3a_se_expand (Conv2D)	(None, 1, 1, 144)	1,008	block3a_se_reduc…
block3a_se_excite (Multiply)	(None, 28, 28, 144)	0	block3a_activati… block3a_se_expan…
block3a_project_co… (Conv2D)	(None, 28, 28, 40)	5,760	block3a_se_excit…
block3a_project_bn (BatchNormalizatio…	(None, 28, 28, 40)	160	block3a_project_…
block3b_expand_conv (Conv2D)	(None, 28, 28, 240)	9,600	block3a_project_…
block3b_expand_bn (BatchNormalizatio…	(None, 28, 28, 240)	960	block3b_expand_c…
block3b_expand_act… (Activation)	(None, 28, 28, 240)	0	block3b_expand_b…
block3b_dwconv	(None, 28, 28,	6,000	

```
block3b_expand_a… |
| (DepthwiseConv2D)   | 240)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_bn          | (None, 28, 28,     |         960 |
block3b_dwconv[0… |
| (BatchNormalizatio… | 240)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_activation  | (None, 28, 28,     |           0 | block3b_bn[0]
[0]  |
| (Activation)        | 240)               |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_se_squeeze  | (None, 240)        |           0 |
block3b_activati… |
| (GlobalAveragePool… |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_se_reshape  | (None, 1, 1, 240)  |           0 |
block3b_se_squee… |
| (Reshape)           |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_se_reduce   | (None, 1, 1, 10)   |       2,410 |
block3b_se_resha… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_se_expand   | (None, 1, 1, 240)  |       2,640 |
block3b_se_reduc… |
| (Conv2D)            |                    |             |
|
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_se_excite   | (None, 28, 28,     |           0 |
block3b_activati… |
| (Multiply)          | 240)               |             |
block3b_se_expan… |
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block3b_project_co… | (None, 28, 28,     |       9,600 |
block3b_se_excit… |
```

```
| (Conv2D)              | 40)                 |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block3b_project_bn    | (None, 28, 28,      |         160 |
block3b_project_… |
| (BatchNormalizatio… | 40)                 |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block3b_drop          | (None, 28, 28,      |           0 |
block3b_project_… |
| (Dropout)             | 40)                 |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block3b_add (Add)     | (None, 28, 28,      |           0 |
block3b_drop[0][… |
|                       | 40)                 |             |
block3a_project_… |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block4a_expand_conv | (None, 28, 28,      |       9,600 |
block3b_add[0][0] |
| (Conv2D)              | 240)                |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block4a_expand_bn     | (None, 28, 28,      |         960 |
block4a_expand_c… |
| (BatchNormalizatio… | 240)                |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block4a_expand_act… | (None, 28, 28,      |           0 |
block4a_expand_b… |
| (Activation)          | 240)                |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block4a_dwconv_pad    | (None, 29, 29,      |           0 |
block4a_expand_a… |
| (ZeroPadding2D)       | 240)                |             |
|                       |                     |             |
├───────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block4a_dwconv        | (None, 14, 14,      |       2,160 |
block4a_dwconv_p… |
| (DepthwiseConv2D)     | 240)                |             |
```

```
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_bn           │ (None, 14, 14,     │         960 │
block4a_dwconv[0… │
│ (BatchNormalizatio… │ 240)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_activation   │ (None, 14, 14,     │           0 │ block4a_bn[0]
[0]   │
│ (Activation)         │ 240)               │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_se_squeeze   │ (None, 240)        │           0 │
block4a_activati… │
│ (GlobalAveragePool… │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_se_reshape   │ (None, 1, 1, 240)  │           0 │
block4a_se_squee… │
│ (Reshape)            │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_se_reduce    │ (None, 1, 1, 10)   │       2,410 │
block4a_se_resha… │
│ (Conv2D)             │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_se_expand    │ (None, 1, 1, 240)  │       2,640 │
block4a_se_reduc… │
│ (Conv2D)             │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_se_excite    │ (None, 14, 14,     │           0 │
block4a_activati… │
│ (Multiply)           │ 240)               │             │
block4a_se_expan… │
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block4a_project_co… │ (None, 14, 14,     │      19,200 │
block4a_se_excit… │
│ (Conv2D)             │ 80)                │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
```

```
──────┤
│ block4a_project_bn  │ (None, 14, 14,     │         320 │
block4a_project_… │
│ (BatchNormalizatio… │ 80)                │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_expand_conv │ (None, 14, 14,     │      38,400 │
block4a_project_… │
│ (Conv2D)            │ 480)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_expand_bn   │ (None, 14, 14,     │       1,920 │
block4b_expand_c… │
│ (BatchNormalizatio… │ 480)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_expand_act… │ (None, 14, 14,     │           0 │
block4b_expand_b… │
│ (Activation)        │ 480)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_dwconv      │ (None, 14, 14,     │       4,320 │
block4b_expand_a… │
│ (DepthwiseConv2D)   │ 480)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_bn          │ (None, 14, 14,     │       1,920 │
block4b_dwconv[0… │
│ (BatchNormalizatio… │ 480)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_activation  │ (None, 14, 14,     │           0 │ block4b_bn[0]
[0]  │
│ (Activation)        │ 480)               │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block4b_se_squeeze  │ (None, 480)        │           0 │
block4b_activati… │
│ (GlobalAveragePool… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| block4b_se_reshape  | (None, 1, 1, 480) |           0 |
block4b_se_squee… |
| (Reshape)           |                   |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_se_reduce   | (None, 1, 1, 20)  |       9,620 |
block4b_se_resha… |
| (Conv2D)            |                   |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_se_expand   | (None, 1, 1, 480) |      10,080 |
block4b_se_reduc… |
| (Conv2D)            |                   |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_se_excite   | (None, 14, 14,    |           0 |
block4b_activati… |
| (Multiply)          | 480)              |             |
block4b_se_expan… |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_project_co… | (None, 14, 14,    |      38,400 |
block4b_se_excit… |
| (Conv2D)            | 80)               |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_project_bn  | (None, 14, 14,    |         320 |
block4b_project_… |
| (BatchNormalizatio… | 80)               |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_drop        | (None, 14, 14,    |           0 |
block4b_project_… |
| (Dropout)           | 80)               |             |
|
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4b_add (Add)   | (None, 14, 14,    |           0 |
block4b_drop[0][… |
|                     | 80)               |             |
block4a_project_… |
├─────────────────────┼───────────────────┼─────────────┼──────────────
──────┤
| block4c_expand_conv | (None, 14, 14,    |      38,400 |
```

```
block4b_add[0][0] │
│ (Conv2D)              │ 480)                │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_expand_bn    │ (None, 14, 14,      │       1,920 │
block4c_expand_c… │
│ (BatchNormalizatio… │ 480)                │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_expand_act… │ (None, 14, 14,      │           0 │
block4c_expand_b… │
│ (Activation)          │ 480)                │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_dwconv       │ (None, 14, 14,      │       4,320 │
block4c_expand_a… │
│ (DepthwiseConv2D)    │ 480)                │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_bn           │ (None, 14, 14,      │       1,920 │
block4c_dwconv[0… │
│ (BatchNormalizatio… │ 480)                │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_activation   │ (None, 14, 14,      │           0 │ block4c_bn[0]
[0]  │
│ (Activation)          │ 480)                │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_se_squeeze   │ (None, 480)         │           0 │
block4c_activati… │
│ (GlobalAveragePool… │                     │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_se_reshape   │ (None, 1, 1, 480)   │           0 │
block4c_se_squee… │
│ (Reshape)             │                     │             │
├──────────────────────┼─────────────────────┼─────────────┼──────────────────
──────┤
│ block4c_se_reduce    │ (None, 1, 1, 20)    │       9,620 │
block4c_se_resha… │
```

(Conv2D)			
block4c_se_expand (Conv2D)	(None, 1, 1, 480)	10,080	block4c_se_reduc…
block4c_se_excite (Multiply)	(None, 14, 14, 480)	0	block4c_activati… block4c_se_expan…
block4c_project_co… (Conv2D)	(None, 14, 14, 80)	38,400	block4c_se_excit…
block4c_project_bn (BatchNormalizatio…	(None, 14, 14, 80)	320	block4c_project_…
block4c_drop (Dropout)	(None, 14, 14, 80)	0	block4c_project_…
block4c_add (Add)	(None, 14, 14, 80)	0	block4c_drop[0][… block4b_add[0][0]
block5a_expand_conv (Conv2D)	(None, 14, 14, 480)	38,400	block4c_add[0][0]
block5a_expand_bn (BatchNormalizatio…	(None, 14, 14, 480)	1,920	block5a_expand_c…

```
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_expand_act… │ (None, 14, 14,     │            0 │
block5a_expand_b… │
│ (Activation)        │ 480)               │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_dwconv      │ (None, 14, 14,     │       12,000 │
block5a_expand_a… │
│ (DepthwiseConv2D)   │ 480)               │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_bn          │ (None, 14, 14,     │        1,920 │
block5a_dwconv[0… │
│ (BatchNormalizatio… │ 480)               │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_activation  │ (None, 14, 14,     │            0 │ block5a_bn[0]
[0]  │
│ (Activation)        │ 480)               │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_se_squeeze  │ (None, 480)        │            0 │
block5a_activati… │
│ (GlobalAveragePool… │                    │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_se_reshape  │ (None, 1, 1, 480)  │            0 │
block5a_se_squee… │
│ (Reshape)           │                    │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_se_reduce   │ (None, 1, 1, 20)   │        9,620 │
block5a_se_resha… │
│ (Conv2D)            │                    │              │
│
├──────────────────────┼────────────────────┼──────────────┼──────────────
──────┤
│ block5a_se_expand   │ (None, 1, 1, 480)  │       10,080 │
block5a_se_reduc… │
│ (Conv2D)            │                    │              │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5a_se_excite    │ (None, 14, 14,       │            0 │
block5a_activati… │
│ (Multiply)           │ 480)                 │              │
block5a_se_expan… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5a_project_co… │ (None, 14, 14,       │      53,760 │
block5a_se_excit… │
│ (Conv2D)             │ 112)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5a_project_bn   │ (None, 14, 14,       │          448 │
block5a_project_… │
│ (BatchNormalizatio… │ 112)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5b_expand_conv │ (None, 14, 14,       │      75,264 │
block5a_project_… │
│ (Conv2D)             │ 672)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5b_expand_bn    │ (None, 14, 14,       │       2,688 │
block5b_expand_c… │
│ (BatchNormalizatio… │ 672)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5b_expand_act… │ (None, 14, 14,       │            0 │
block5b_expand_b… │
│ (Activation)         │ 672)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5b_dwconv       │ (None, 14, 14,       │      16,800 │
block5b_expand_a… │
│ (DepthwiseConv2D)    │ 672)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
──────┤
│ block5b_bn           │ (None, 14, 14,       │       2,688 │
block5b_dwconv[0… │
│ (BatchNormalizatio… │ 672)                 │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────
```

```
──────┤
│ block5b_activation  │ (None, 14, 14,     │           0 │ block5b_bn[0]
[0]  │
│ (Activation)        │ 672)               │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_se_squeeze  │ (None, 672)        │           0 │
block5b_activati… │
│ (GlobalAveragePool… │                    │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_se_reshape  │ (None, 1, 1, 672)  │           0 │
block5b_se_squee… │
│ (Reshape)           │                    │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_se_reduce   │ (None, 1, 1, 28)   │      18,844 │
block5b_se_resha… │
│ (Conv2D)            │                    │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_se_expand   │ (None, 1, 1, 672)  │      19,488 │
block5b_se_reduc… │
│ (Conv2D)            │                    │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_se_excite   │ (None, 14, 14,     │           0 │
block5b_activati… │
│ (Multiply)          │ 672)               │             │
block5b_se_expan… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_project_co… │ (None, 14, 14,     │      75,264 │
block5b_se_excit… │
│ (Conv2D)            │ 112)               │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block5b_project_bn  │ (None, 14, 14,     │         448 │
block5b_project_… │
│ (BatchNormalizatio… │ 112)               │             │
│                     │                    │             │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| block5b_drop         | (None, 14, 14,     |           0 |
block5b_project_… |
| (Dropout)            | 112)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5b_add (Add)    | (None, 14, 14,     |           0 |
block5b_drop[0][… |
|                      | 112)               |             |
block5a_project_… |
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_expand_conv | (None, 14, 14,     |     75,264 |
block5b_add[0][0] |
| (Conv2D)             | 672)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_expand_bn    | (None, 14, 14,     |      2,688 |
block5c_expand_c… |
| (BatchNormalizatio… | 672)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_expand_act… | (None, 14, 14,     |           0 |
block5c_expand_b… |
| (Activation)         | 672)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_dwconv       | (None, 14, 14,     |     16,800 |
block5c_expand_a… |
| (DepthwiseConv2D)    | 672)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_bn           | (None, 14, 14,     |      2,688 |
block5c_dwconv[0… |
| (BatchNormalizatio… | 672)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_activation   | (None, 14, 14,     |           0 | block5c_bn[0]
[0]   |
| (Activation)         | 672)               |             |
|
├──────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
| block5c_se_squeeze   | (None, 672)        |           0 |
```

```
block5c_activati… │
│ (GlobalAveragePool… │                   │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_se_reshape   │ (None, 1, 1, 672) │           0 │
block5c_se_squee… │
│ (Reshape)            │                   │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_se_reduce    │ (None, 1, 1, 28)  │      18,844 │
block5c_se_resha… │
│ (Conv2D)             │                   │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_se_expand    │ (None, 1, 1, 672) │      19,488 │
block5c_se_reduc… │
│ (Conv2D)             │                   │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_se_excite    │ (None, 14, 14,    │           0 │
block5c_activati… │
│ (Multiply)           │ 672)              │              │
block5c_se_expan… │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_project_co… │ (None, 14, 14,    │      75,264 │
block5c_se_excit… │
│ (Conv2D)             │ 112)              │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_project_bn   │ (None, 14, 14,    │         448 │
block5c_project_… │
│ (BatchNormalizatio… │ 112)              │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_drop         │ (None, 14, 14,    │           0 │
block5c_project_… │
│ (Dropout)            │ 112)              │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block5c_add (Add)    │ (None, 14, 14,    │           0 │
block5c_drop[0][… │
```

```
|                       | 112)                |             |
block5b_add[0][0] |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_expand_conv | (None, 14, 14,     |     75,264 |
block5c_add[0][0] |
| (Conv2D)             | 672)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_expand_bn   | (None, 14, 14,     |      2,688 |
block6a_expand_c… |
| (BatchNormalizatio… | 672)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_expand_act… | (None, 14, 14,     |          0 |
block6a_expand_b… |
| (Activation)         | 672)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_dwconv_pad  | (None, 17, 17,     |          0 |
block6a_expand_a… |
| (ZeroPadding2D)      | 672)                |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_dwconv      | (None, 7, 7, 672) |     16,800 |
block6a_dwconv_p… |
| (DepthwiseConv2D)   |                     |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_bn          | (None, 7, 7, 672) |      2,688 |
block6a_dwconv[0… |
| (BatchNormalizatio… |                     |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_activation  | (None, 7, 7, 672) |          0 | block6a_bn[0]
[0]  |
| (Activation)         |                     |             |
|
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6a_se_squeeze  | (None, 672)         |          0 |
block6a_activati… |
| (GlobalAveragePool… |                     |             |
```

```
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6a_se_reshape  │ (None, 1, 1, 672)  │           0 │
block6a_se_squee… │
│ (Reshape)           │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6a_se_reduce   │ (None, 1, 1, 28)   │      18,844 │
block6a_se_resha… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6a_se_expand   │ (None, 1, 1, 672)  │      19,488 │
block6a_se_reduc… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6a_se_excite   │ (None, 7, 7, 672)  │           0 │
block6a_activati… │
│ (Multiply)          │                    │             │
block6a_se_expan… │
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6a_project_co… │ (None, 7, 7, 192)  │     129,024 │
block6a_se_excit… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6a_project_bn  │ (None, 7, 7, 192)  │         768 │
block6a_project_… │
│ (BatchNormalizatio… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6b_expand_conv │ (None, 7, 7,       │     221,184 │
block6a_project_… │
│ (Conv2D)            │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
────────┤
│ block6b_expand_bn   │ (None, 7, 7,       │       4,608 │
block6b_expand_c… │
│ (BatchNormalizatio… │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────
```

```
──────┤
│ block6b_expand_act… │ (None, 7, 7,       │           0 │
block6b_expand_b… │
│ (Activation)        │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_dwconv      │ (None, 7, 7,       │      28,800 │
block6b_expand_a… │
│ (DepthwiseConv2D)   │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_bn          │ (None, 7, 7,       │       4,608 │
block6b_dwconv[0… │
│ (BatchNormalizatio… │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_activation  │ (None, 7, 7,       │           0 │ block6b_bn[0]
[0]  │
│ (Activation)        │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_squeeze  │ (None, 1152)       │           0 │
block6b_activati… │
│ (GlobalAveragePool… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_reshape  │ (None, 1, 1,       │           0 │
block6b_se_squee… │
│ (Reshape)           │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_reduce   │ (None, 1, 1, 48)   │      55,344 │
block6b_se_resha… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6b_se_expand   │ (None, 1, 1,       │      56,448 │
block6b_se_reduc… │
│ (Conv2D)            │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

block6b_se_excite (Multiply)	(None, 7, 7, 1152)	0	block6b_activati… block6b_se_expan…
block6b_project_co… (Conv2D)	(None, 7, 7, 192)	221,184	block6b_se_excit…
block6b_project_bn (BatchNormalizatio…	(None, 7, 7, 192)	768	block6b_project_…
block6b_drop (Dropout)	(None, 7, 7, 192)	0	block6b_project_…
block6b_add (Add)	(None, 7, 7, 192)	0	block6b_drop[0][… block6a_project_…
block6c_expand_conv (Conv2D)	(None, 7, 7, 1152)	221,184	block6b_add[0][0]
block6c_expand_bn (BatchNormalizatio…	(None, 7, 7, 1152)	4,608	block6c_expand_c…
block6c_expand_act… (Activation)	(None, 7, 7, 1152)	0	block6c_expand_b…
block6c_dwconv	(None, 7, 7,	28,800	

```
block6c_expand_a… │
│ (DepthwiseConv2D)   │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_bn          │ (None, 7, 7,       │       4,608 │
block6c_dwconv[0… │
│ (BatchNormalizatio… │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_activation  │ (None, 7, 7,       │           0 │ block6c_bn[0]
[0]  │
│ (Activation)        │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_se_squeeze  │ (None, 1152)       │           0 │
block6c_activati… │
│ (GlobalAveragePool… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_se_reshape  │ (None, 1, 1,       │           0 │
block6c_se_squee… │
│ (Reshape)           │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_se_reduce   │ (None, 1, 1, 48)   │      55,344 │
block6c_se_resha… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_se_expand   │ (None, 1, 1,       │      56,448 │
block6c_se_reduc… │
│ (Conv2D)            │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_se_excite   │ (None, 7, 7,       │           0 │
block6c_activati… │
│ (Multiply)          │ 1152)              │             │
block6c_se_expan… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block6c_project_co… │ (None, 7, 7, 192)  │     221,184 │
block6c_se_excit… │
```

```
| (Conv2D)              |                     |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6c_project_bn   | (None, 7, 7, 192) |         768 |
block6c_project_… |
| (BatchNormalizatio… |                     |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6c_drop         | (None, 7, 7, 192) |           0 |
block6c_project_… |
| (Dropout)             |                     |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6c_add (Add)    | (None, 7, 7, 192) |           0 |
block6c_drop[0][… |
|                       |                     |             |
block6b_add[0][0] |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6d_expand_conv | (None, 7, 7,       |     221,184 |
block6c_add[0][0] |
| (Conv2D)              | 1152)               |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6d_expand_bn    | (None, 7, 7,       |       4,608 |
block6d_expand_c… |
| (BatchNormalizatio… | 1152)               |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6d_expand_act… | (None, 7, 7,       |           0 |
block6d_expand_b… |
| (Activation)          | 1152)               |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6d_dwconv       | (None, 7, 7,       |      28,800 |
block6d_expand_a… |
| (DepthwiseConv2D)    | 1152)               |             |
|                       |                     |             |
├──────────────────────┼─────────────────────┼─────────────┼──────────────
──────┤
| block6d_bn           | (None, 7, 7,       |       4,608 |
block6d_dwconv[0… |
| (BatchNormalizatio… | 1152)               |             |
```

```
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_activation  │ (None, 7, 7,       │           0 │ block6d_bn[0]
[0]   │
│ (Activation)        │ 1152)              │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_se_squeeze  │ (None, 1152)       │           0 │
block6d_activati… │
│ (GlobalAveragePool… │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_se_reshape  │ (None, 1, 1,       │           0 │
block6d_se_squee… │
│ (Reshape)           │ 1152)              │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_se_reduce   │ (None, 1, 1, 48)   │      55,344 │
block6d_se_resha… │
│ (Conv2D)            │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_se_expand   │ (None, 1, 1,       │      56,448 │
block6d_se_reduc… │
│ (Conv2D)            │ 1152)              │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_se_excite   │ (None, 7, 7,       │           0 │
block6d_activati… │
│ (Multiply)          │ 1152)              │             │
block6d_se_expan… │
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_project_co… │ (None, 7, 7, 192)  │     221,184 │
block6d_se_excit… │
│ (Conv2D)            │                    │             │
│
├──────────────────────┼────────────────────┼─────────────┼──────────────
───────┤
│ block6d_project_bn  │ (None, 7, 7, 192)  │         768 │
block6d_project_… │
│ (BatchNormalizatio… │                    │             │
│
```

```
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block6d_drop         │ (None, 7, 7, 192)    │            0 │
block6d_project_… │
│ (Dropout)            │                      │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block6d_add (Add)    │ (None, 7, 7, 192)    │            0 │
block6d_drop[0][… │
│                      │                      │              │
block6c_add[0][0] │
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block7a_expand_conv  │ (None, 7, 7,         │      221,184 │
block6d_add[0][0] │
│ (Conv2D)             │ 1152)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block7a_expand_bn    │ (None, 7, 7,         │        4,608 │
block7a_expand_c… │
│ (BatchNormalizatio… │ 1152)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block7a_expand_act… │ (None, 7, 7,         │            0 │
block7a_expand_b… │
│ (Activation)         │ 1152)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block7a_dwconv       │ (None, 7, 7,         │       10,368 │
block7a_expand_a… │
│ (DepthwiseConv2D)    │ 1152)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block7a_bn           │ (None, 7, 7,         │        4,608 │
block7a_dwconv[0… │
│ (BatchNormalizatio… │ 1152)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
──────┤
│ block7a_activation   │ (None, 7, 7,         │            0 │ block7a_bn[0]
[0]  │
│ (Activation)         │ 1152)                │              │
│
├──────────────────────┼──────────────────────┼──────────────┼──────────────────
```

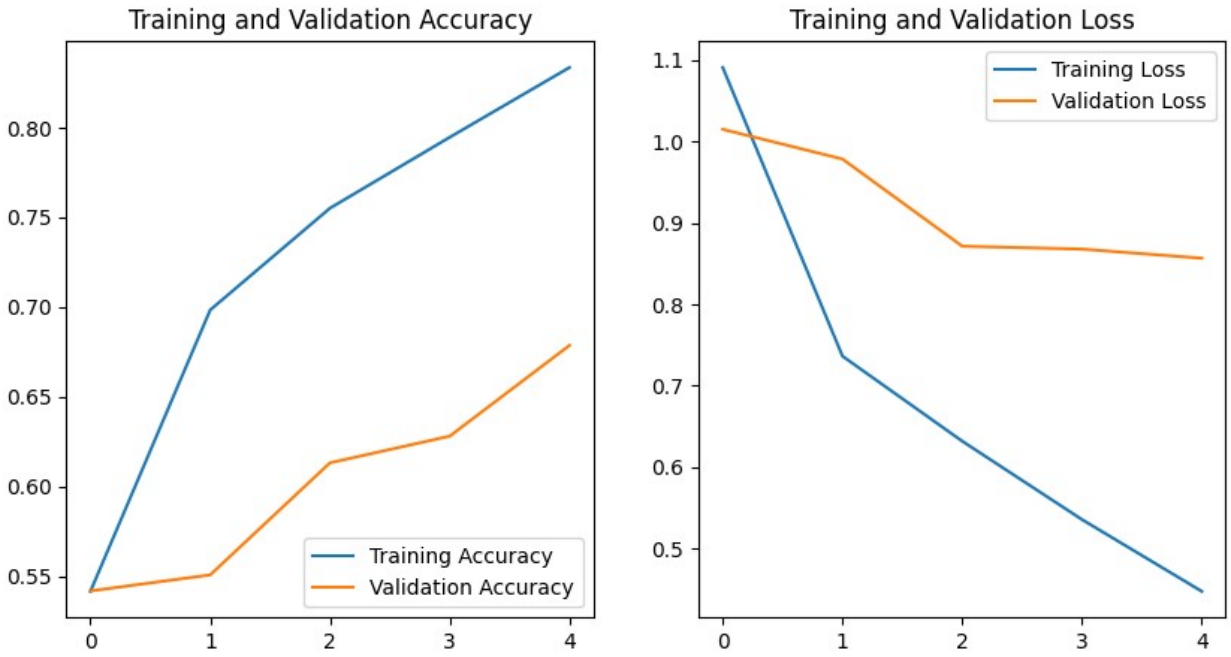
```
──────┤
│ block7a_se_squeeze  │ (None, 1152)       │           0 │
block7a_activati… │
│ (GlobalAveragePool… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block7a_se_reshape  │ (None, 1, 1,       │           0 │
block7a_se_squee… │
│ (Reshape)           │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block7a_se_reduce   │ (None, 1, 1, 48)   │      55,344 │
block7a_se_resha… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block7a_se_expand   │ (None, 1, 1,       │      56,448 │
block7a_se_reduc… │
│ (Conv2D)            │ 1152)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block7a_se_excite   │ (None, 7, 7,       │           0 │
block7a_activati… │
│ (Multiply)          │ 1152)              │             │
block7a_se_expan… │
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block7a_project_co… │ (None, 7, 7, 320)  │     368,640 │
block7a_se_excit… │
│ (Conv2D)            │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ block7a_project_bn  │ (None, 7, 7, 320)  │       1,280 │
block7a_project_… │
│ (BatchNormalizatio… │                    │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
│ top_conv (Conv2D)   │ (None, 7, 7,       │     409,600 │
block7a_project_… │
│                     │ 1280)              │             │
│
├─────────────────────┼────────────────────┼─────────────┼──────────────────
──────┤
```

```
| top_bn                | (None, 7, 7,       |      5,120 | top_conv[0]
[0]    |
| (BatchNormalizatio… | 1280)              |            |
|
├──────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| top_activation       | (None, 7, 7,       |          0 | top_bn[0][0]
|
| (Activation)         | 1280)              |            |
|
├──────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| global_average_poo… | (None, 1280)       |          0 |
top_activation[0… |
| (GlobalAveragePool… |                    |            |
|
├──────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| dense_5 (Dense)      | (None, 1024)       |  1,311,744 |
global_average_p… |
├──────────────────────┼────────────────────┼────────────┼──────────────
──────┤
| dense_6 (Dense)      | (None, 4)          |      4,100 | dense_5[0][0]
|
└──────────────────────┴────────────────────┴────────────┴──────────────
──────┘

 Total params: 5,365,415 (20.47 MB)

 Trainable params: 1,315,844 (5.02 MB)

 Non-trainable params: 4,049,571 (15.45 MB)
```

```
# Compile the model
EfficientNetB0model.compile(optimizer=Adam(learning_rate=0.001),
              loss='sparse_categorical_crossentropy',  # Suitable for
multi-class classification
              metrics=['accuracy'])

# Fit the model
epochs=20
history = EfficientNetB0model.fit(train_ds,
validation_data=validation_ds, epochs=epochs, callbacks =
[early_stopping, reduce_lr] )
```

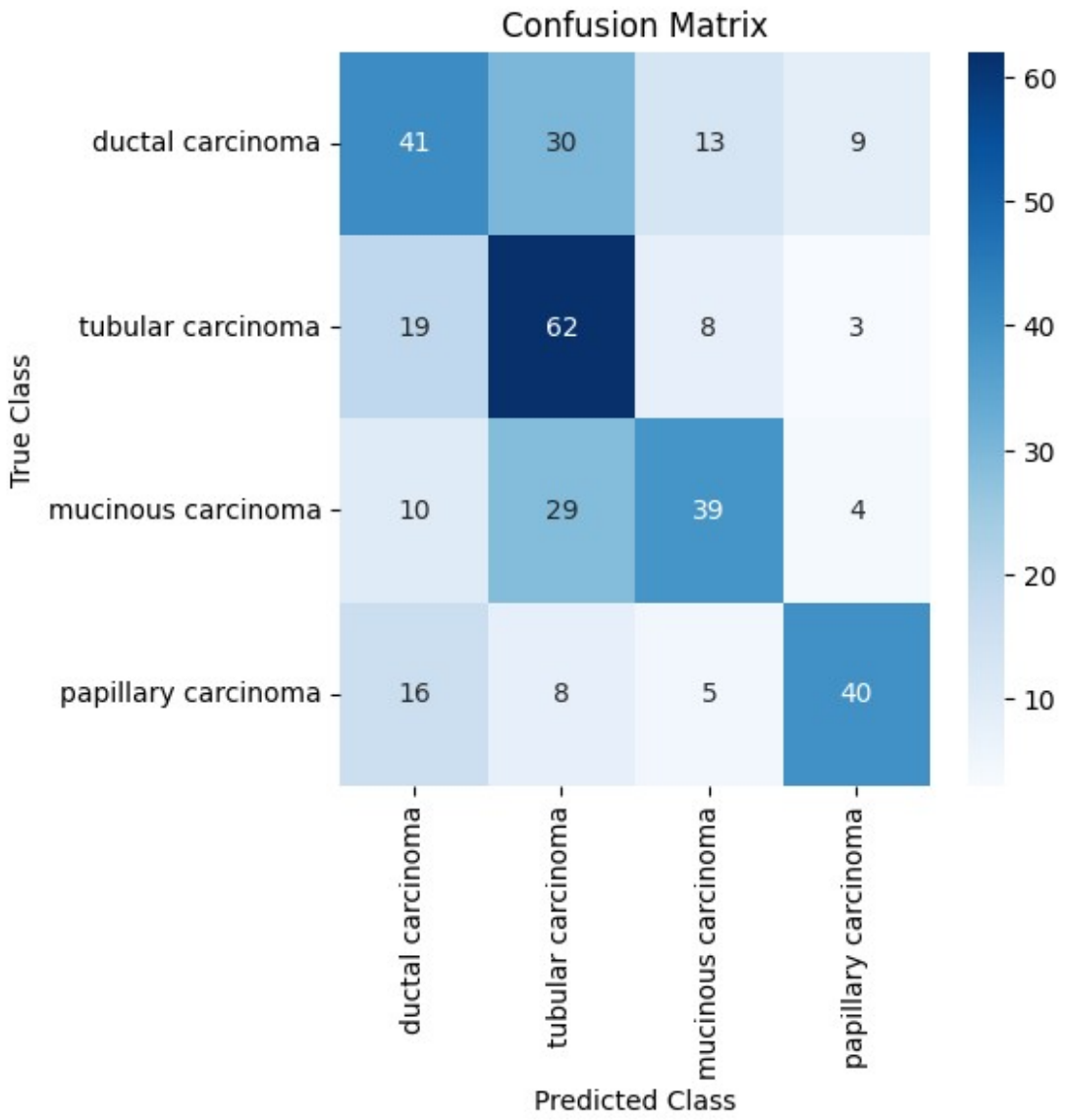
```
Epoch 1/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 91s 1s/step - accuracy: 0.4835 - loss:
1.2516 - val_accuracy: 0.5417 - val_loss: 1.0149 - learning_rate:
0.0010
Epoch 2/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 60s 1s/step - accuracy: 0.6864 - loss:
```

```
0.7555 - val_accuracy: 0.5506 - val_loss: 0.9784 - learning_rate:
0.0010
Epoch 3/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 53s 949ms/step - accuracy: 0.7621 - loss:
0.6133 - val_accuracy: 0.6131 - val_loss: 0.8713 - learning_rate:
0.0010
Epoch 4/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 52s 932ms/step - accuracy: 0.7901 - loss:
0.5528 - val_accuracy: 0.6280 - val_loss: 0.8678 - learning_rate:
0.0010
Epoch 5/20
56/56 ━━━━━━━━━━━━━━━━━━━━ 52s 928ms/step - accuracy: 0.8332 - loss:
0.4610 - val_accuracy: 0.6786 - val_loss: 0.8566 - learning_rate:
0.0010
Epoch 5: early stopping
Restoring model weights from the end of the best epoch: 1.

EfficientNetB0_acc = history.history['accuracy']
EfficientNetB0_val_acc = history.history['val_accuracy']

EfficientNetB0_loss = history.history['loss']
EfficientNetB0_val_loss = history.history['val_loss']

epochs_range = range(len(history.history['accuracy']))

plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, EfficientNetB0_acc, label='Training Accuracy')
plt.plot(epochs_range, EfficientNetB0_val_acc, label='Validation
Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, EfficientNetB0_loss, label='Training Loss')
plt.plot(epochs_range, EfficientNetB0_val_loss, label='Validation
Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```




```
# Evaluate the model on the validation/test dataset
EfficientNetB0_val_loss, EfficientNetB0_val_accuracy =
EfficientNetB0model.evaluate(validation_ds)
print(f'EfficientNetB0 Validation Loss: {EfficientNetB0_val_loss}')
print(f'EfficientNetB0 Validation Accuracy:
{EfficientNetB0_val_accuracy}')

# Predict the classes on the validation/test dataset
EfficientNetB0_Y_pred = EfficientNetB0model.predict(validation_ds)
EfficientNet_y_pred = EfficientNetB0_Y_pred.argmax(axis=1)  # Get the
index of the max probability (predicted class)

12/12 ━━━━━━━━━━━━━━━━━━━━ 10s 842ms/step - accuracy: 0.5580 - loss:
1.0079
EfficientNetB0 Validation Loss: 1.014897346496582
EfficientNetB0 Validation Accuracy: 0.5416666865348816
12/12 ━━━━━━━━━━━━━━━━━━━━ 28s 877ms/step

#3) EfficientNetB0
#  Check if y_true needs to be converted
if y_true.ndim == 1:
    EfficientNetB0_y_true_classes = y_true  # If it's already a vector
of class labels
else:
    EfficientNetB0_y_true_classes = np.argmax(y_true, axis=1)  # If
it's one-hot encoded

# y_pred_classes is already computed correctly
EfficientNetB0_y_pred_classes = np.argmax(EfficientNetB0_Y_pred,
```

```
axis=1)  # Predicted class labels

# Compute the confusion matrix
conf_matrix = confusion_matrix(EfficientNetB0_y_true_classes,
EfficientNetB0_y_pred_classes)

# Plot the confusion matrix
# Plot the confusion matrix
class_names = ['ductal carcinoma', 'tubular carcinoma', 'mucinous
carcinoma', 'papillary carcinoma']

plt.figure(figsize=(5, 5))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues',
            xticklabels=class_names, yticklabels=class_names)
plt.title('Confusion Matrix')
plt.xlabel('Predicted Class')
plt.ylabel('True Class')
plt.show()
```




```
# predicted probabilities from the model
y_pred_prob = EfficientNetB0model.predict(X_test)

# Step 2: Convert y_test to one-hot encoding (if not already done)
y_test_one_hot = to_categorical(y_test, num_classes = 4)

# Step 3: Calculate the ROC AUC score for multi-class classification
roc_auc = roc_auc_score(y_test_one_hot, y_pred_prob,
multi_class='ovr')
print(f"ROC AUC: {roc_auc}")

11/11 ━━━━━━━━━━━━━━━━━━━━ 24s 2s/step
ROC AUC: 0.8513146323969191
```

```python
# Classification Report
#3) EfficientNetB0
report = classification_report(EfficientNetB0_y_true_classes,
EfficientNetB0_y_pred_classes, target_names=class_names)
print('Classification Report:')
print(report)
```

```
Classification Report:
                     precision    recall  f1-score   support

   ductal carcinoma       0.48      0.44      0.46        93
  tubular carcinoma       0.48      0.67      0.56        92
 mucinous carcinoma       0.60      0.48      0.53        82
papillary carcinoma       0.71      0.58      0.64        69

           accuracy                           0.54       336
          macro avg       0.57      0.54      0.55       336
       weighted avg       0.56      0.54      0.54       336
```

```python
#Look at how you can change the averages (weighted etc.) so that the
average is not overly represented by the bigger classs
```