COVID-19 DETECTION USING CONVOLUTIONAL NEURAL NETWORK

Review Paper

A Project Work

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE and ENGINEERING

With Specialization in

Artificial Intelligence and Machine Learning

Submitted by:

MANASIJ HALDAR

20BCS6838

Under the Supervision of:

PRABHJOT SINGH MANOCHA

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING APEX INSTITUE OF TECHNOLOGY

CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413, PUNJAB

MAY,2022

DECLARATION

I, **Manasij Haldar**, student of **Bachelor of Engineering in Computer Science and Engineering with Specialization in Artificial Intelligence and Machine Learning**, **session:2020-2024**, Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, hereby declare that the work presented in this Project Work entitled '**Covid-19 Detection using Convolutional Neural Network'** is the outcome of our own bona fide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Manasij Haldar
Candidate UID:20BCS6838

Date: 4/20/2022
Place: Chandigarh University, Mohali, Punjab, India

Abstract

In early December 2019, A novel severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) was responsible for the Coronavirus Disease 2019 (COVID-19) outbreak in Wuhan, Hubei Province, China. On January 30, 2020, the World Health Organization designated the outbreak as a Public Health Emergency of International Concern. The emergence of COVID-19 in 2020 has been a historical moment. This virus has spread to many countries and rendered many people housebound. Many studies have attempted to examine the impact of this pandemic from various angles; however, this study will focus on how it has affected and may affect children aged 0 to 12 years in the future after schools have been closed for months. As of February 14, 2020, 49,053 laboratory-confirmed cases and 1,381 deaths had been reported worldwide. In response to the perceived risk of contracting disease, many countries have implemented a variety of control measures.

We conducted a literature review of publicly available information to summarize what we know about the virus and the current epidemic. This literature review covers the causal agent, pathogenesis, and immunological responses, epidemiology, diagnosis, treatment, and management of the disease, as well as control and prevention efforts.

The Coronavirus Disease 2019 (COVID-19) pandemic is still wreaking havoc on the global population's health and well-being. Effective screening of infected patients is a critical step in the fight against COVID-19, with radiology examination using chest radiography being one of the key screening approaches. Early studies discovered that patients with COVID-19 infection have abnormalities in chest radiography images. In this study, we introduce a deep convolutional neural network design tailored for the detection of COVID-19 cases from chest X-ray (CXR) images, motivated by this and inspired by the research community's open-source efforts. For the training, we used various open source CXR datasets

available on the internet to create a larger and more efficient dataset that is now uploaded and available on Kaggle. We present an open access benchmark dataset that we created with around 10,000 CXR images and, to the best of the authors' knowledge, the largest number of publicly available COVID-19 positive cases.

In addition, we presented a comparison study of how different transfer learning models make predictions using an explainable method in an attempt to gain deeper insights into critical factors associated with COVID cases, which can aid clinicians in better screening, as well as to validate that it is making decisions based on relevant information from the CXR images. The goal of the study is to provide a better understanding for future COVID-19 or other imagery-based predictive decision-making tasks in terms of selecting the best transfer learning model to achieve the highest accuracy in the operations.

Keywords

- 2019-nCoV
- COVID-19 Outbreak
- SARS-CoV-2
- Novel coronavirus
- Convolutional Neural Network
- Deep Learning
- Transfer Learning
- Python
- Kaggle
- Model Training
- Model Testing

Table of Contents

1. INTRODUCTION

1.1 A Brief Background

Throughout the first months of 2020, the new SARS-CoV-2 coronavirus, which causes COVID-19, had the entire world on edge. It caused many nations' borders to close and millions of inhabitants to be confined to their homes due to sick persons, with 868,000 confirmed cases worldwide at this time (April 2020). In December of this year, the virus was discovered in China. Since March 2020, Europe has been the epicenter of the virus's spread.

China has managed to contain the virus over three months after the outbreak began in December 2019, with a total of 3,312 deaths and more than 81,000 afflicted persons. According to a report released in April 2020, Italy, which overtook the Asian countries in death toll in March 2020, has become the most impacted country, with more than 10,000 people killed. This number was steadily increasing. Various studies projected the expansion of infection curves based on various parameters such as the number of exposed, infected, and recovered humans. These investigations helped researchers to gain a better understanding of the transmission dynamics that may occur in each country.

The outbreak's cause has yet to be determined. The first cases were discovered in December of this year. Respiratory symptoms, fever, cough, dyspnea, and viral pneumonia are all clinical features of COVID-19. The fundamental issue with these symptoms is that there are virus-infected persons who are asymptomatic.

COVID-19 is detected through the collection of samples from the respiratory tract. When the case study is asymptomatic or symptoms are light, it is performed at home by a health care expert, or in a health facility or hospital if the patient is admitted for a serious ailment. In nations like Germany and South Korea, performing as many tests as possible has proven to be the most effective way to combat the virus.

Because Spain was unable to conduct so many tests, it is critical to explore and create alternate techniques for doing these tests in a timely and efficient manner.

In the detection and follow-up of disease, AI and radiomics applied to X-Ray and Computed Tomography (CT) are useful tools. According to CT scans, significant ground grass opacity lesions in the peripheral and posterior lungs indicate COVID-19 pneumonia. As a result, once abnormalities in chest radiographs are suggestive of coronavirus, CT can play a crucial role in the diagnosis of COVID-19 as an advanced imaging evidence. AI algorithms and radiomics features generated from chest X-rays would be extremely useful in implementing large-scale screening programmes in any country with X-ray capability and would aid in the diagnosis of COVID-19.

The current circumstance necessitates the implementation of an automatic detection system as an alternative diagnosis option to stop COVID-19 from spreading. Several researchers have used machine learning to accomplish this goal, such as the Size Aware Random Forest approach (iSARF) proposed by, in which participants were divided into groups with varying sizes of infected lesions. Then, with each group, a random forest-based classifier was trained. Under five-fold cross-validation, their proposed technique generated an accuracy of 0.879, a sensitivity of 0.907, and a specificity of 0.833, according to the results.

In order to produce better outcomes than more typical machine learning approaches, deep learning techniques are also applied. Convolutional neural networks are one of the most widely utilized approaches in picture categorization (CNNs). This type of model has been used in several studies to detect COVID-19 in medical images, such as in, where the authors propose a CNN model trained with a randomly selected set of image regions of interest (ROIs), achieving an accuracy of 85.2 percent, a specificity of 0.83, and a sensitivity of 0.67. Another example of what may be accomplished with CNNs is offered by. They propose the COVID-Net CNN network, which achieves 92.4 percent accuracy, 80% sensitivity, and 88.9%

specificity. COVIDX-Net is a method that combines seven distinct deep convolutional neural network designs, including a modified version of the Visual Geometry Group Network (VGG19) and the second edition of Google MobileNet. Each deep neural network model can categorize the patient's status as a negative or positive COVID-19 case by analyzing the normalized intensities of the X-ray image. For healthy and COVID-19 detection, their trials reach f1-scores of 0.89 and 0.91, respectively.

The findings in the cited papers suggest that deep learning techniques are beneficial for virus identification and that they improve the metrics acquired using more typical machine learning methods

Our paper's main contribution is to increase COVID-19 detection accuracy by offering a new dataset that combines COVID-19 and pneumonia images to create more consistent predictions and by using image processing to allow image normalization and improve model learning.

1.2 Problem Definition

The reference diagnostic test for COVID-19 pneumonia is real-time reverse transcription-polymerase chain reaction (RT-PCR). The specificity of RT-PCR is approximately 95%, but the sensitivity of RT-PCR at the initial presentation is 60% to 71% because of kit performance, sampling, and transportation limitations. Because of these low sensitivity rates and the need for rapid diagnosis, X-Ray has been frequently used in the current pandemic condition. Also, several cases with initial negative RT-PCR results are reported to have positive chest CT findings or X-rays. So, in the direction of finding an accurate way of testing Covid19, X-rays tend to be more trustworthy than any RT-PCR reports, blood reports or various other symptoms. Which led us here for the project of Covid19 Detection analyzing X-Ray imagery using Neural Networks.

1.3 Project Overview

In this project, we aim to predict two classes outcome from the input fed Image data of a patient's Chest X-Ray, to determine whether the patient is suffering from COVID-19 or not. We plan to use Transfer Learning, under the domain of Deep Learning using Convolutional Neural Network to proceed with the project. We are to create a model with a very large dataset, split into Train Test and Validation segments. Four different Transfer Learning models are proposed to be used on the dataset and the outcome is to be on the basis of the most efficient model out of them.

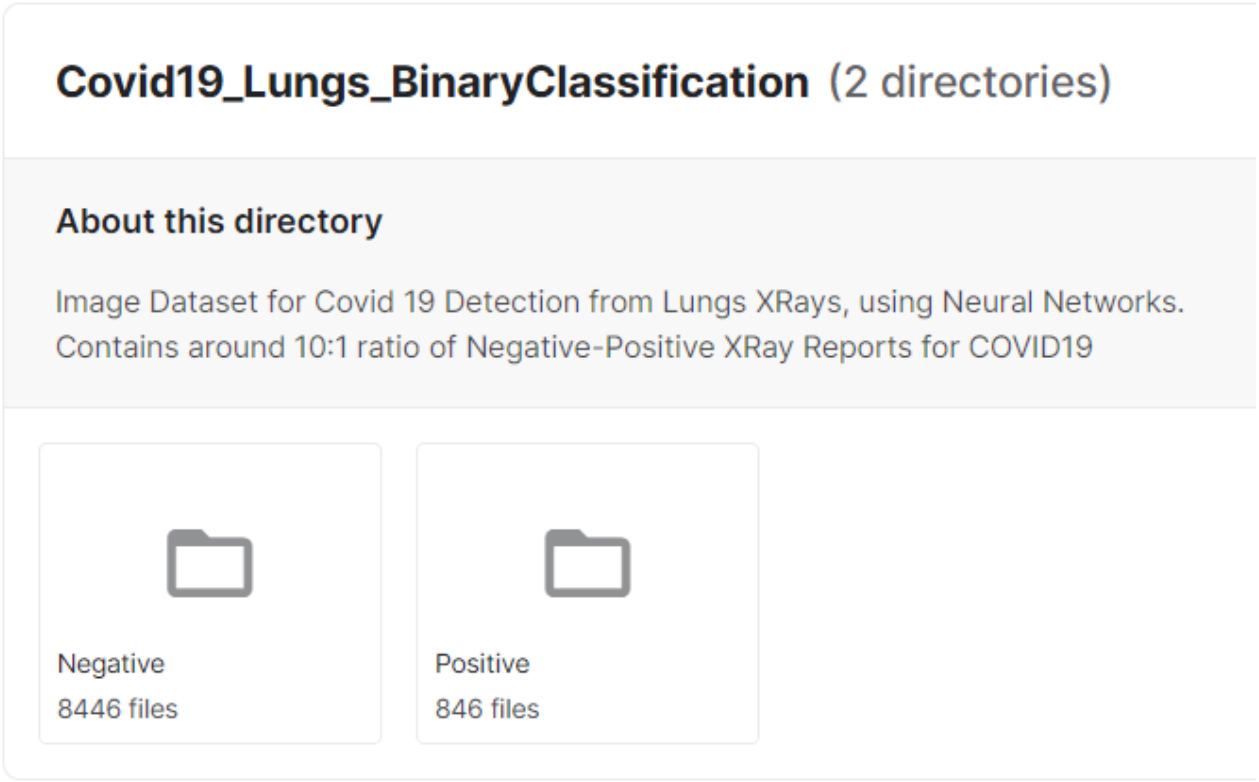
1.4 Software Specification

The Project is carried on Python 3.7 kernel with environments of Jupyter Notebook on Kaggle, Google Colab, and PyCharm IDE.

Important Python Libraries were used such as:
Pandas (for Data Processing),
Tensorflow-Keras(for transfer learning and deep learning models and tools), **OS, Shutil** (for Dataset manipulations),
Matplotlib, Seaborn(For Data and Outcome Visualization)
IPython(for Image Processing)

1.5 Data Specification

Various Datasets were collected from open-source platforms comprising of Positive and Negative Covid CXR images. The entire data is cleaned and processed to form a workable dataset containing around 10000 images of both classes.



The Dataset is publicly uploaded and available on Kaggle Datasets.

2. LITERATURE REVIEW

CXRs are a good monitor of COVID-19 chest manifestations, and its scoring system provides an accurate method to predict the disease severity. The study also revealed a positive correlation between the patients' age and total severity score to the final disease outcome providing a good indicator for clinician to identify at an early stage the patients with the highest risk and plan specific treatment strategies for them.

COVID-19 is a highly infectious disease that has been spread widely throughout the world. The disease management strategies primarily depend upon the early disease diagnosis. However, the dramatic dissemination of the disease created a great challenge due to the insufficient laboratory kits. That is why radiology has become a forefront method during the outbreak of COVID-19.

Current literature is mostly assessing COVID-19 CT findings, as it offers more

sensitive results than chest X-ray (CXR) especially in the initial assessment of the patients. The increased number of hospitalized patients and the consequent increase in radiological examinations would make the constant use of chest CT scan (from diagnosis to discharge) difficult to sustain over time. The dependence on CT creates a huge burden on radiology departments and this makes the CXRs greatly substitute the CT examinations. Although chest X-ray (CXR) is considered less sensitive for the detection of pulmonary involvement in early-stage disease, it is useful for monitoring the rapid progression of lung abnormalities in COVID-19[8] [9], especially in critical patients admitted to intensive care units. To provide valuable help for the clinicians and improve the stratification of the disease risk, chest X-ray (CXR) scoring system was tailored providing a semi-quantitative tool for assessment of lung abnormalities.

In this study, we analyzed the CXRs findings and severity scores of patients proven to have COVID-19 in different stages of disease. CXRs abnormalities were detected in 268 of 350 patients (77%) at certain points of the disease course. In our study, each lung was given a score of 0–4 depending on the extent of lung involvement (score 0 = no involvement; 1 ≤ 25%; 2 = 25–50%; 3 = 50–75%; 4 ≥ 75% lung affection). A total severity score was calculated by summing both lung scores (total severity scores ranged from 0 to 8). Borghesi et al. made another CXR scoring system for COVID-19 pneumonia (Brixia score) by dividing the lungs to six zones on frontal projection (upper, middle, and lower zones); then, a score (from 0 to 3) is assigned to each zone based on the lung abnormalities detected on frontal chest projection as follows: score 0, no lung abnormalities; score 1, interstitial infiltrates; score 2, interstitial and alveolar infiltrates (interstitial predominance); and score 3, interstitial and alveolar infiltrates (alveolar predominance). The scores of the six lung zones are then added to obtain an overall “CXR SCORE” ranging from 0 to 18 . In our study, most of the patients showed bilateral lung affection (181 patients, 67.5%) with lower zonal predominance (196, 73.1%) and peripheral distribution (156 patients, 58.2%). The most common CXRs features detected in COVID-19

cases were consolidation seen in 218 patients (81.3%), followed by reticular interstitial thickening seen in 107 patients (39.9%) and GGO seen in 87 patients (32.5%). Few cases showed pulmonary nodules seen in 25 patients (9.3%) and pleural effusion seen in 20 patients (7.5%). This agreed with Wong et al. who did a study on 64 COVID-19 patients, they found that Consolidation was the most common finding (47%), followed by GGO (33%). Also, peripheral predominance was seen in 41% of CXR abnormalities with lower zone distribution (50%) with bilateral lung involvement (50%). Pleural effusion was uncommon, only seen in 3%. Also, Lomoro et al. performed a study on thirty-two patients of COVID-19 disease; they found that consolidation is the most common finding (46.9%) with bilateral lung infection in (78.1%) and lower zone involvement (52%). No pleural effusion was identified. Jacobi et al. stated that standard CXR can easily identify reticular opacities accompanying regions of ground glass attenuation. They state that air space consolidation opacities with peripheral and lower zone distribution are unique for COVID-19 disease. Chen et al. reported bilateral pneumonia as the most common finding on chest radiographs. While Ng et al. reported that CXR lacks sensitivity in the early stages of lung disease. In most studies, pleural effusions, pneumothorax, and lung cavitation are rare in COVID-19 infected patients. Pneumothorax was detected in 2 cases in our study, and it was iatrogenic due to mechanical ventilation in intubated patients. We classified patients according to the stage of illness into four stages (1–4 days, 5–9 days, 10–15 days, and > 15 days). The degree of disease severity was assessed using a semi-quantitative CXRs severity score that reflects the severity of different stages of this disease. The total severity score was lowest at stage 1 compared to other stages, with significant difference among other stages, indicating that the disease changed rapidly within 10–15 days after the onset of the initial symptoms. We estimated the total severity score in the baseline and follow-up CXR, and it ranged from 0 to 8. In most cases (230 patients, 65.7%), TSS was mild, ranging between 0 and 2. While, in 82 patients (23.4%), there was a moderate severity score ranging between 3 and 5. Severe cases with a severity score of between 6 and 8 was found in 38 patients (10.9%) with more

disseminated lung involvement. Wong et al. found in their study that 41% had mild findings with a total severity score of 1–2, while moderate and severe cases with more extensive lung involvement were seen in 20% and 8% patients, who had severity scores of 3–4 and 5–6, respectively. There was no patient who had a severity score of > 6 on their baseline CXR with the severity of CXR findings peaking at 10–15 days from the date of symptom onset. In our study, the maximum total severity score was reached in 113 patients (42.2%) in the initial baseline CXR with mean total severity score 1.49 ± 1.53 followed by 92 patients (34.3%) who reached the maximum TSS at 1st follow-up CXR done (done 1–4 days) with mean total severity score 2.08 ± 1.83. The highest total severity score of the CXR findings was found in the 4th follow-up CXR 15 days after the onset of the symptoms with its mean 4.51 ± 1.61. Our study correlated the disease outcome to the patients' age with a significant difference between the age of the patients and COVID-19 disease outcome (P value = 0.008). The mean age for the recovered patients was 41.09 ± 14.14 while the mean age for the dead patients was 51.04 ± 10.17. Lowest mortality rate was observed in 20–40 years, while patients aging 40–59 and ≥ 60 years showed significantly higher mortality rate. In our study, there were 261 males (74.6%) and 89 females (25.4%) with male patients showing significantly higher mortality rate compared to the female patients (P value 0.025). This agreed with Borghesi et al., who did a study on 783 Italian patients. They found that most patients (67.9%) were males and only 15.2% were younger than 50 years. They stated that in older age groups between 50 and 79 years, there was more significant pulmonary infection with highest severity score seen in males ≥ 50 years or female ≥ 80 especially that underlying comorbidities (such as hypertension, diabetes, cardiovascular disease, and oncologic history) are risk factors of fatal outcome in adult patients with confirmed SARS-CoV-2 infection. In our study, the disease outcome showed a positive correlation with the maximum severity score (6.87 ± 0.71 for the dead patients and 2.06 ± 1.84 for the surviving patients) with high statistical significance (P value < 0.001). In patients with TSS 2, there was a statistical significance between the TSS and the outcome of COVID disease for the survived patients (P value

0.032), while, in patients with TSS 7 and 8, there was a highly statistical significance for the outcome for the dead patients (P value < 0.001). This agreed with Toussie et al. that showed that the severity of lung involvement on the initial chest radiograph was associated with more need for patients' hospitalization as well as the increased risk of intubation and have proposed the use of initial CXR severity scores as a prognostic indicator of COVID-19 patients' outcome. The major strength of this study is the large sample size, which consisted of 350 COVID-19 patients. Our study had some limitations. First, it is a retrospective analysis. Second, the lack of correlation between CXR severity score and patient comorbidities (such as hypertension, diabetes, cardiovascular disease, and oncologic history). Third, not all the patients could be followed till the final outcome as the course of the disease was truncated in these patients. Fourth, CXR serial follow-up studies were not performed in a uniform pattern as it was dedicated by the clinician as regards the clinical condition. Fifth, for severe cases in the intensive care unit, the portable AP CXR was suboptimal with only few cases performed CT, so we could not judge the sensitivity of CXR.

3. PROBLEMFORMULATION

Background

During a pandemic, such as COVID-19, a timely and precise diagnosis is critical. It improves patient outcomes and relieves burden on health-care systems that are dealing with a growing rate of infection.

The polymerase chain reaction (PCR) is the current preferred approach for diagnosing COVID-19 (PCR). However, some of the worst-affected locations are unable to obtain enough kits to fulfill demand, and many countries are unable to conduct tests due to a lack of lab facilities.

Deep learning models, which are a type of artificial intelligence (AI), are being studied and used to detect and diagnose a wide range of diseases.

Deep learning algorithms could be employed in this case to identify infected individuals using chest X-ray scans, which are readily available around the world. This approach could be employed in situations where PCR diagnostics are currently unavailable.

Deep learning for X-ray analysis might drastically cut the time it takes to diagnose patients, with an AI model processing up to 200 images in the time it takes a radiologist to analyze one.

Method

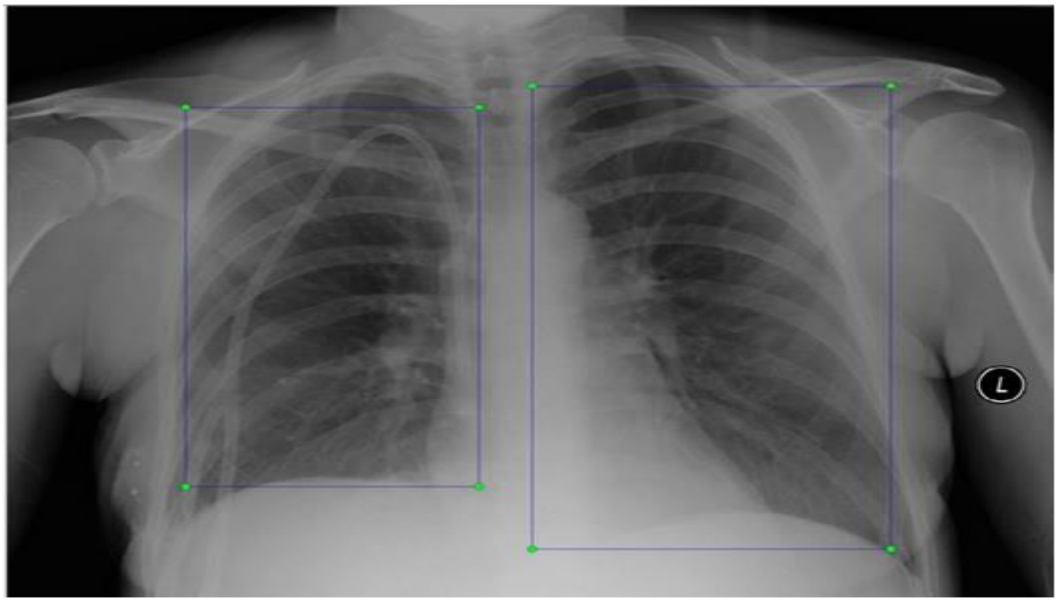
We propose using a deep convolutional neural network trained on COVID-19 and pneumonia images, as well as a fresh dataset comprising COVID-19 and pneumonia images. Both are open to the public via GitHub and Kaggle, respectively. COVID-19 cases are represented by the chest X-ray or CT images available on GitHub. It was made by putting together medical photos from publicly accessible sources and publications. There are 204 COVID-19 X-ray pictures in this dataset. The Kaggle dataset, on the other hand, was produced for a pneumonia detection challenge. Bounding boxes surround sick lung regions in the photos. Without the boundary boxes, the samples are negative and show no signs of pneumonia. The presence of bounding boxes in samples indicates the presence of pneumonia.

By combining COVID-19 and pneumonia images, we propose a new dataset that is both larger and more diversified. Because normal pneumonia and COVID-19 have comparable appearances in chest X-ray images, including pneumonia images in the training dataset implies an extra advantage. This dataset merger enables the creation of a more robust model capable of distinguishing between those diseases. Another benefit of this merge is that it expands the train dataset, which is important because COVID-19 photos are scarce at the time of authoring this research. Because of the

similarities between pneumonia and COVID-19, this merge does not increase the size of the COVID-19 picture collection, but it does improve detection quality.

To avoid biased findings, we split the photos into train and test sets, dividing all the data in a balanced way, meaning that all samples of each class in the training sets are well-balanced. Despite the fact that we have a significant number of pneumonia and normal photos, we created a dataset of 1500 photographs for this reason.

For training, we chose 104 COVID-19 images, 205 health lung images, and 204 pneumonia images, and for testing, we chose 100 COVID-19 images, 444 health lung images, and 443 pneumonia images. There are more samples of pneumonia and normal photos in the Kaggle dataset, but we chose only 205 for training to ensure a balanced dataset. We add more pneumonia and normal photos to the test stage to demonstrate the model's robustness in detecting COVID-19 with no false positives. To summarize, we employ a total of 513 photos in the train set and 887 images in the test set.

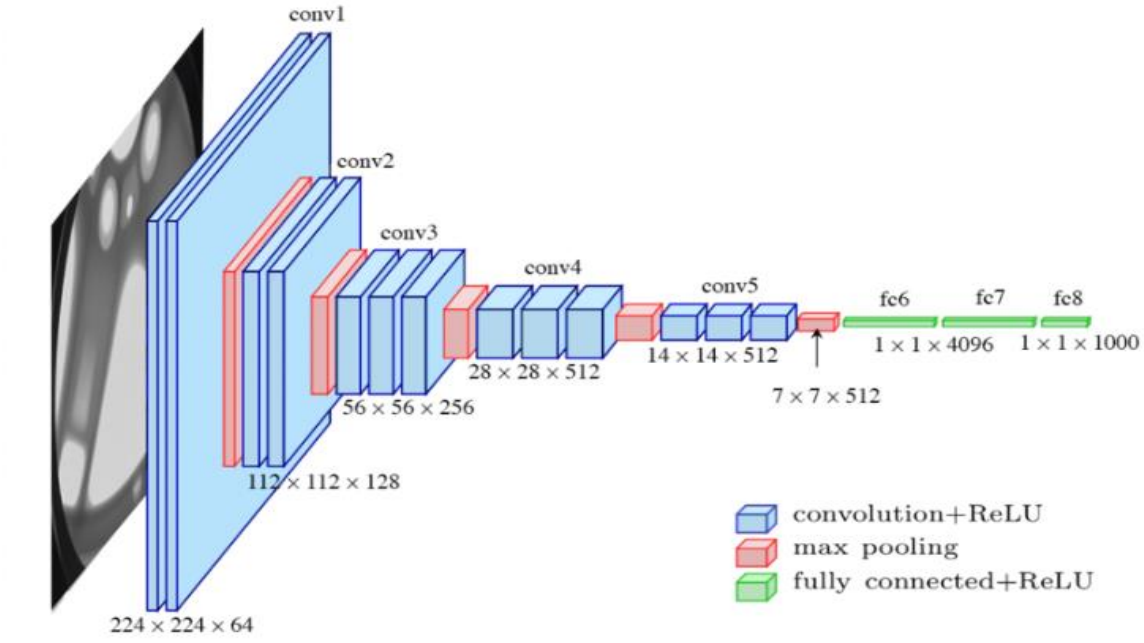


4. MODEL ARCHITECTURE

The employed architecture was chosen based on the good results gained with CNNs in state-of-the-art works for COVID-19 image classification, as well as the good results obtained with this kind of architecture in other related tasks. Based on the

Single Shot Multibox Detector, we adopted the same network design provided in (SSD). Using a single deep neural network, this design is optimized for recognizing objects in photos. This method divides the output space of bounding boxes into a series of default boxes for each feature map point, using varying aspect ratios and sizes. The network generates scores for the presence of each item type in each default box at prediction time, and then adjusts the box to better match the object shape.

SSD offers comparable accuracy to approaches that use more than one architecture for detecting objects, while being faster and providing a single framework for both training and inference, according to experimental results on various notable datasets. SSD offers substantially better accuracy than other single-stage algorithms, even with a smaller input image size.



In this design, we employ VGG-16 as the basis network for feature extraction. Fast R-CNN is also used in this model. We have many boxes with varied sizes and aspect ratios over the entire image during training. When compared to the ground truth, SSD determines the box with the highest Intersection-Over-Union (IoU)

Our main goal is to develop a more robust model that can handle a wide range of input item sizes and forms. As a result, a data augmentation step is conducted during the SSD training. The following operations are applied to each image in the dataset as part of this process:

• Keep the original input image in its entirety.

• Pick a patch with a minimum overlap of 0.1, 0.3, 0.5, 0.7, or 0.9 with the objects. Each sampled patch is a proportion of the original image size between [0.1, 1].

• Pick a patch at random.

5. MODEL COMPARISON REVIEW

Before finalizing the Model with VGG-16, a few other models were also trained and tested with the same dataset for a comparative study on the accuracy on each of them.

For the study, the following pre-built models were used from the keras.applications library:

- DenseNet169
- VGG16
- InceptionV3
- ResNet152V2

Details on the models are provided for a better idea.

Inception Model:

Introduction:

In 2014, Google (together with other academic institutions) published a paper describing a novel deep learning convolutional neural network architecture, which

at the time was the largest and most efficient deep neural network architecture available.

An Inception Network was used as the unique design, and a GoogLeNet version won the ImageNet Large-scale Visual Recognition Challenge 2014 classification computer vision competition with the best performance (ILVRC14).

Since 2014, both we and the deep learning field have come a long way. Several deep learning architectures will approach or exceed human-level classification and object detection performance by 2020.

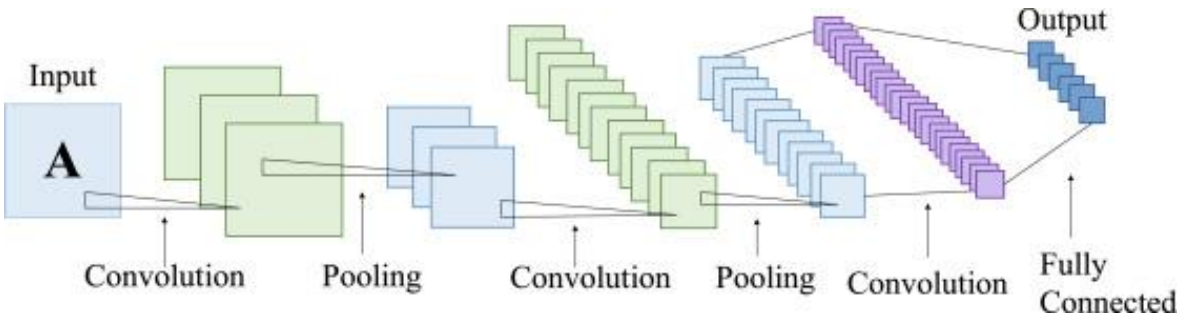
On the other hand, convolutional neural networks owe their innovations and advancements to their forefathers.

Architecture of Inception Model:

An inception network is a deep neural network with Inception modules, which are repeating architectural designs.

The following are the essential principles to remember when developing an Inception model:

1. High performance necessitates the use of large deep neural networks. To be considered huge, a neural network must include multiple more layers and units inside these levels.
2. Convolutional neural networks benefit from extracting features at various scales. The biological human visual brain recognizes patterns at different scales, which are then combined to create larger object perceptions. As a result, multi-scale networks are able to learn more.
3. Consider the Hebbian Principle, which claims that neurons that fire at the same time will connect.



TYPE	PATCH / STRIDE SIZE	INPUT SIZE
Conv	3×3/2	299×299×3
Conv	3×3/1	149×149×32
Conv padded	3×3/1	147×147×32
Pool	3×3/2	147×147×64
Conv	3×3/1	73×73×64
Conv	3×3/2	71×71×80
Conv	3×3/1	35×35×192
3 × Inception	Module 1	35×35×288
5 × Inception	Module 2	17×17×768
2 × Inception	Module 3	8×8×1280
Pool	8 × 8	8 × 8 × 2048
Linear	Logits	1 × 1 × 2048
Softmax	Classifier	1 × 1 × 1000

In practical cases, the above-mentioned guidelines have several technological flaws. In large networks, overfitting is prevalent, and chaining numerous convolutional methods together increases the network's computing cost. To make the Inception network and module a reality, the researchers designed intuitive convnet topologies.

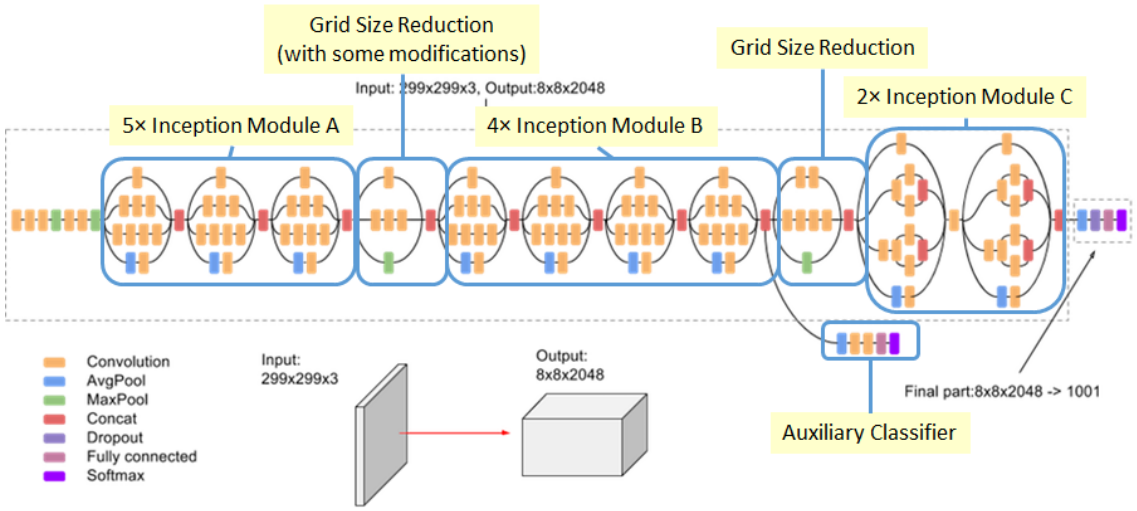
In this 2013 report, Lin et al. proposed 1x1 convolution, popularly known as

"Network in Network." A 1x1 convolution uses the element-wise product of all pixel values in an image. The image (input data) is convoluted with the conv 1x1 filter, resulting in an output with the dimensions 1 x 1 x n (where 'n' is the number of filters). A 1x1 filter does not learn spatial patterns inside the image, but it does learn patterns across the depth of the image (cross channel). As a result, 1x1 convolution filters not only decrease method dimension but also allow the network to learn more.

The 1x1 convolution reduces the number of input channels, which results in fewer channels in the output. This section is the Inception network's bottleneck layer (shown in a diagram further down below). Pooling layers down-sample images as they go through the network (lower the height and width). 1x1 convolution has the added benefit of reducing the image's height, width, and number of channels.

To improve the performance of a neural network, increase the number of layers (depth) and units/neuron within the layers (width), which ineffectively develops a more comprehensive network.

Structure of a basic Inception model is given below:



The Inception network has the luxury of using different filter sizes within its convolutional layers. The 1x1 convolution, one of the convolution filter sizes used

by Inception, has already been addressed. The others are 3x3 and 5x5.

Within a convnet, different conv filter sizes learn spatial patterns and recognize properties at different scales.

Prior to the advent of the Inception network, researchers had to choose filter sizes to use in deep convolutional neural networks to obtain optimal performance.

By utilizing different filter sizes of 1x1, 3x3, and 5x5, Inception eliminates the need for such decisions. 1x1 learns patterns across the depth of the input, whereas 3x3 and 5x5 learn spatial patterns across the three dimensions of the input (height, width, and depth). The representational capacity rises when all of the patterns acquired from the various filter sizes are combined.

To give a single Inception module output, the Inception module contains a concatenation layer that merges all of the conv filters' outputs and feature maps into a single object.

Benefits of the Inception Module:

1. Convolutional neural networks provide a high level of performance gain.
2. Efficient computing resource utilization for an Inception network's high-performance output with minimum increase in compute load
3. Extraction of features from input data at various scales using different convolutional filter sizes
4. Cross-channel patterns are learned by 1x1 conv filters, allowing the network to extract more information.

ResNet

Introduction:

ResNet, or Residual Networks, is a well-known neural network that is used to perform a variety of computer vision tasks. This model was the ImageNet challenge winner in 2015. ResNet was a game-changer because it allowed us to train 150-layer deep neural networks. Prior to ResNet, training very deep neural networks was challenging due to the problem of vanishing gradients.

AlexNet, the ImageNet 2012 winner and the model that seems to have spurred interest in deep learning, has just eight convolutional layers, compared to 19 for the VGG network, 22 for Inception or GoogleNet, and 152 for ResNet 152.

Architecture of ResNet:

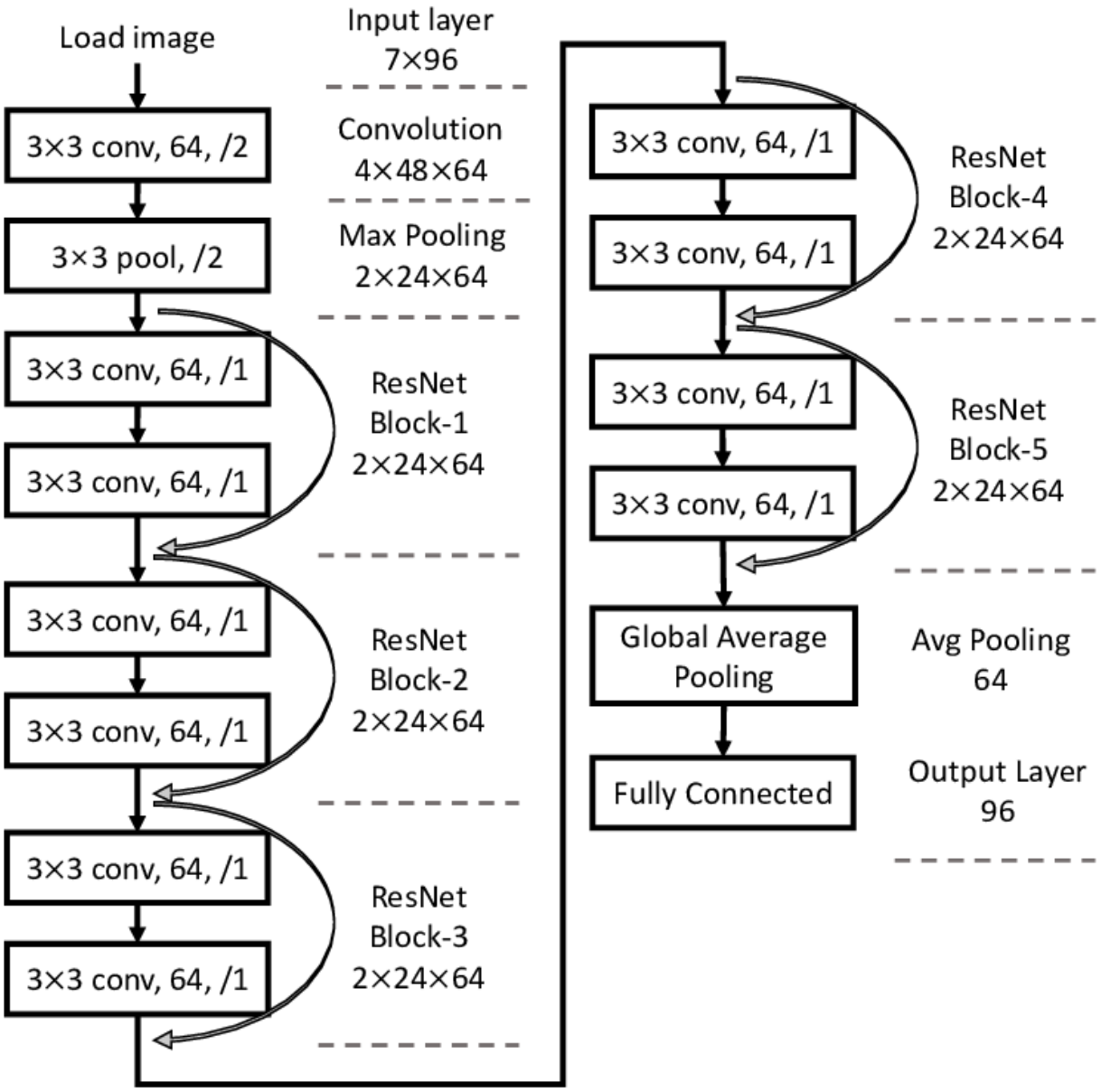
ResNet is a Convolutional Neural Network (CNN) architecture that tackles the "vanishing gradient" problem, enabling deeper networks to outperform shallower networks. A decreasing gradient happens during backpropagation. When there are too many layers in the neural network training process, the gradient becomes very small until it disappears, and optimization is halted. The problem is resolved by ResNet adopting "identity shortcut connections."

It has two stages of implementation:

ResNet generates and bypasses a large number of levels that aren't used initially, recycling activation functions from preceding layer.

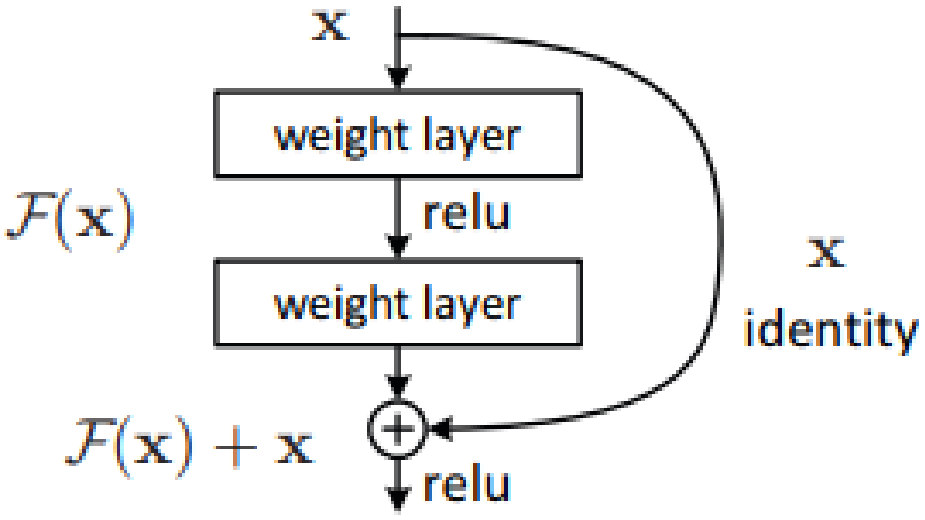
A second stage involves retraining the network and extending the "residual" convolutional layers. This enables the investigation of more parts of the feature space that a shallow convolutional network architecture would have skipped.

Structure of a basic ResNet model is given below:



ResNet was the first to introduce the concept of skip connection. The graphic below depicts the skip connection. Convolution layers are piled one on top of the other in the figure on the left. Convolution layers are still stacked on the right, but the original input is now added to the convolution block's output. This is known as "skipping connection."

These Residual blocks were introduced to help with the issue of training very deep networks, and the ResNet model is made up of them.



The introduction of these Residual blocks alleviated the challenge of training very deep networks, and the ResNet model is built up of these blocks. The first thing we note in the above diagram is that there is a direct connection that skips several of the model's levels. The heart of residual blocks is a connection known as the ‘skip connection.' Because of the skip connection, the output is not same. Without the skip connection, input 'X is multiplied by the layer's weights, then a bias term is added. Then comes the activation function, f() and we get the output as H(x).

H(x)=f(wx + b) or H(x)=f(x)

Thanks to the installation of a new skip connection method, the output is now H(x), where

H(x)=f(x)+x

However, when using a convolutional layer or pooling layers, the input dimension may differ from the output dimension. As a result, these two ways can be used to solve the problem:

- To enhance its dimensions, zero is padded with the skip connection.
- To match the dimensions, 11 convolutional layers are added to the input.

In this situation, the result is:

$$H(x)=f(x)+w1.x$$

In this case, an additional parameter w1 is added, but in the first technique, no additional parameter is added.
ResNet's skip connections strategy tackles the problem of disappearing gradient in deep CNNs by enabling the gradient to flow along an additional shortcut channel. In addition, if any layer degrades architecture performance, regularization will skip it.

The architecture is inspired on VGG-19 and has a 34-layer plain network to which shortcut and skip connections are added. These residual blocks or skip connections change the design into a residual network.

Advantages of ResNet:

1. Networks with a large number of layers (even thousands) can be easily taught without increasing the percentage of training errors.
2. Using identity mapping, ResNets can help solve the vanishing gradient problem.

DenseNet:

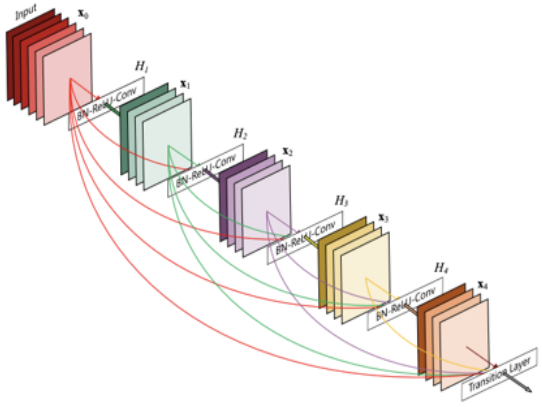
Introduction:

The Dense Convolutional Network, or simply DenseNet, is an innovative neural network discovery for visual object detection. DenseNet is quite similar to ResNet, although there are several key distinctions. DenseNet concatenates (.) the output of the previous layer with the output of the future layer, whereas ResNet utilizes an additive approach (+) that combines the previous layer (identity) with the future layer. DenseNet was created primarily to address the vanishing gradient's effect on

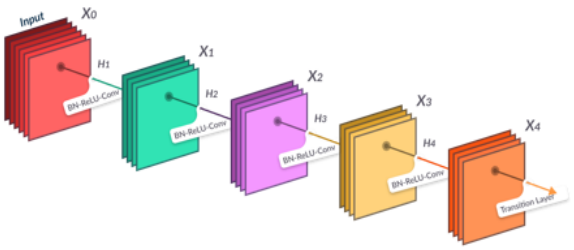
high-level neural networks' accuracy. Simply said, the information evaporates before reaching its target due to the longer travel between the input and output layers. It’s a convolutional neural network that use Dense Blocks to create dense connections between layers, with all layers (with matching feature-map sizes) connected directly to one another. To retain the feed-forward nature, each layer gets extra inputs from all preceding levels and passes on its own feature-maps to all subsequent layers.

Architecture:

Dense Convolutional Network (DenseNet) is a feed-forward network that connects each layer to every other layer. Our network has L(L+1)/2 direct connections, whereas standard convolutional networks with L layers have L connections - one between each layer and its succeeding layer. All previous layers' feature maps are utilized as inputs into each layer, and its own feature maps are used as inputs into all subsequent layers. DenseNets have a number of compelling advantages, including the elimination of the vanishing-gradient problem, improved feature propagation, feature reuse, and a significant reduction in the number of parameters.



DenseNet Structure



ResNet Structure

$$a^{[l]} = g\left([a^{[0]}, a^{[1]}, a^{[2]}, \dots\dots\dots\dots, a^{[l-1]}\right)$$

$$a^{[l]} = g\left(z^{[l+1]} + a^{[l]}\right)$$

Layers	Output Size	DenseNet 169
Convolution	112×112	7×7 conv, stride 2
Pooling	56×56	3×3 max pool, stride 2
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv
	28×28	2×2 average pool, stride 2
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv
	14×14	2×2 average pool, stride 2
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$
Transition Layer (3)	14×14	1×1 conv
	7×7	2×2 average pool, stride 2
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$
Classification Layer	1×1	7×7 global average pool
	1000	1000D fully-connected, softmax

The standard ResNet structure and a 5-layer dense block with a growth rate of k = 4are shown in this figure. Using the composite function operation, the previous layer's output becomes the second layer's input. Convolution, pooling, batch normalization, and non-linear activation layers are all part of this composite operation.

The network has L(L+1)/2 direct connections because of these links. The architecture has L layers.

DenseNet comes in a variety of flavors, including DenseNet-121, DenseNet-160, and DenseNet-201. The numbers represent the neural network's layer count.

DenseNet (Dense Convolutional Network) is a network design that focuses on making deep learning networks grow deeper while also making them more efficient to train by employing shorter connections between layers. DenseNet is a

convolutional neural network in which each layer is connected to all other layers deeper in the network, so the first layer is connected to the 2nd, 3rd, 4th, and so on. This is done to allow maximal information flow between network tiers. Each layer takes inputs from all previous levels and passes on its own feature maps to all subsequent layers to maintain the feed-forward character. Unlike ResNets, it concatenates features instead of summarizing them. The 'i^{th}' layer, then, contains I inputs and is made up of feature maps from all the convolutional blocks before it. All of the following 'I-i' layers receive its own feature maps. In contrast to standard deep learning designs, this introduces '(I(I+1))/2' connections to the network. As a result, it has fewer parameters than standard convolutional neural networks because no meaningless feature maps need to be learned.

Apart from the basic convolutional and pooling layers, DenseNet has two key components. Dense Blocks and Transition layers are what they're called.

DenseNet begins with a basic layer of convolution and pooling. Then there's a dense block followed by a transition layer, followed by another dense block followed by a transition layer, followed by another dense block followed by a transition layer, and lastly a dense block followed by a classification layer.

The first convolution block contains 64 7x7 filters with a stride of 2. A MaxPooling layer with 3x3 max pooling and a stride of 2 follows.

Each convolutional block passes through the following phases after receiving the input: BatchNormalization, ReLU activation, and lastly the actual Conv2D layer. Every dense block has two convolutions, with kernel sizes of 1x1 and 3x3. This is repeated six times in dense block 1, twelve times in dense block 2, twenty-four times in dense block 3, and sixteen times in dense block four.

Each of the 1x1 convolutions in dense block has four times the number of filters. As a result, we use four filters, but three of them are only present once. In addition, the input and output tensors must be concatenated.

Advantages of DenseNet:

They solve the vanishing-gradient problem, improve feature propagation, promote feature reuse, and cut the number of parameters in half. In comparison to alternative systems, the DenseNet architecture has numerous notable advantages. The authors first claim that their architecture outperforms the other competing architectures in ImageNet. My research in Near-Identical Images confirmed that the DenseNet design indeed provide the optimum image representation. Second, the authors claim that their increased parameter efficiency makes it easier to train the network. This is true when compared to other network topologies of equal size. I believe that the training time is competitive with that of certain lower-layer networks. The improved performance is well worth the additional training time.

VGG16:

Introduction:

In their publication "Very Deep Convolutional Networks for Large-Scale Image Recognition," K. Simonyan and A. Zisserman from the University of Oxford proposed the VGG16 convolutional neural network model. In ImageNet, a dataset of over 14 million images belonging to 1000 classes, the model obtains 92.7 percent top-5 test accuracy. It was a well-known model that was submitted to the ILSVRC-2014. It outperforms AlexNet by sequentially replacing big kernel-size filters (11 and 5 in the first and second convolutional layers, respectively) with numerous 33 kernel-size filters. VGG16 had been training for weeks on NVIDIA Titan Black GPUs.

VGG is a multilayer deep Convolutional Neural Network (CNN) architecture that stands for Visual Geometry Group. VGG-16 and VGG-19 feature 16 and 19 convolutional layers, respectively, and the term "deep" refers to the number of layers.

Cutting-edge object recognition models are built on top of the VGG architecture. On a range of applications and datasets other than ImageNet, the VGGNet, which was constructed as a deep neural network, beats baselines. It is also one of the most commonly utilized image recognition architectures today. The VGGNet-16 contains 16 layers and can categorize pictures into 1000 different object categories, including keyboards, animals, pens, and mice. The model also accepts images with a resolution of 224 by 224 pixels.

Architecture:

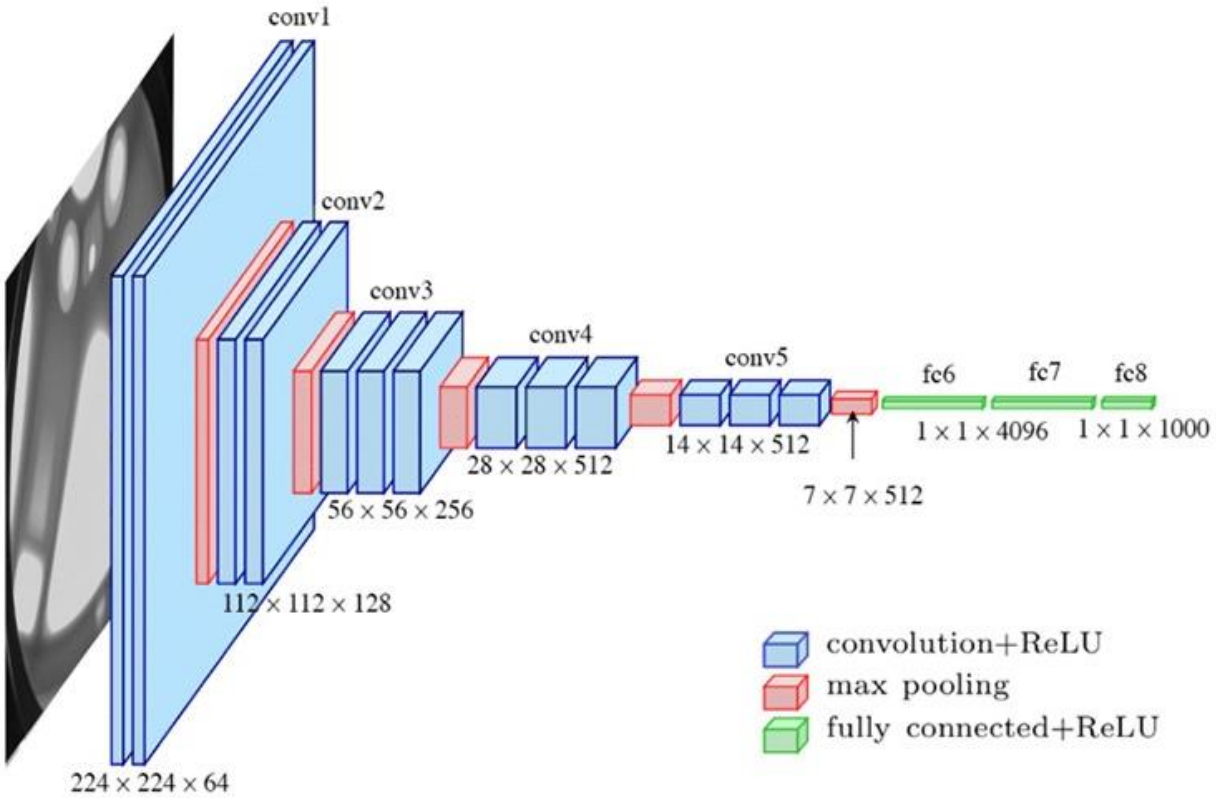
Small convolutional filters are used to build the VGG network. There are 13 convolutional layers and three fully linked layers in the VGG-16.

Let's look at VGG's architecture in more detail:

- **Input:** The VGGNet accepts images with a size of 224x224 pixels. To keep the image input size consistent for the ImageNet competition, the model's authors chopped out the center 224x224 patch in each image.
- **Convolutional Layers**: VGG's convolutional layers use a small receptive field (3x3), the smallest size that still captures up/down and left/right movement. There are also 1x1 convolution filters that perform a linear transformation of the input. Then there's a ReLU unit, which is a significant AlexNet invention that cuts training time in half. The rectified linear unit activation function (ReLU) is a piecewise linear function that outputs the input if it is positive and zero otherwise. To maintain spatial resolution after

convolution, the convolution stride is set to 1 pixel (stride is the number of pixel shifts over the input matrix).

- **Hidden Layers:** The VGG network's hidden layers all use ReLU. Local Response Normalization (LRN) is rarely used in VGG since it increases memory usage and training time. Furthermore, it has no effect on total accuracy.
- **Fully Connected Layers:** There are three fully connected layers in the VGGNet. The first two layers each have 4096 channels, whereas the third layer has 1000 channels, one for each class.



Layer		Feature Map	Size	Kernel Size	Stride	Activation
Input	Image	1	224 x 224 x 3	-	-	-
1	2 X Convolution	64	224 x 224 x 64	3x3	1	relu
	Max Pooling	64	112 x 112 x 64	3x3	2	relu
3	2 X Convolution	128	112 x 112 x 128	3x3	1	relu
	Max Pooling	128	56 x 56 x 128	3x3	2	relu
5	2 X Convolution	256	56 x 56 x 256	3x3	1	relu
	Max Pooling	256	28 x 28 x 256	3x3	2	relu
7	3 X Convolution	512	28 x 28 x 512	3x3	1	relu
	Max Pooling	512	14 x 14 x 512	3x3	2	relu
10	3 X Convolution	512	14 x 14 x 512	3x3	1	relu
	Max Pooling	512	7 x 7 x 512	3x3	2	relu
13	FC	-	25088	-	-	relu
14	FC	-	4096	-	-	relu
15	FC	-	4096	-	-	relu
Output	FC	-	1000	-	-	Softmax

The number 16 in the term VGG alludes to the deep neural network's 16 layers (VGGNet). This indicates that VGG16 is a large network with over 138 million parameters. Even by modern standards, it is a massive network. The simplicity of the VGGNet16 architecture, on the other hand, is what makes the network appealing. It may be argued that its architecture is quite uniform just by glancing at it.

A few convolution layers are followed by a pooling layer that decreases the height and width of the image. When it comes to the number of filters we can employ, we have roughly 64 options, which we can expand to around 128 and then to 256. We can utilize 512 filters in the final levels.

Every step or stack of the convolution layer doubles the number of filters that can be used. This is a key design principle for the VGG16 network's architecture. One of the major disadvantages of the VGG16 network is that it is a large network, which means that training its parameters takes longer.

The VGG16 model is over 533MB in size because to its depth and number of fully connected layers. Implementing a VGG network is hence time-consuming.

Although the VGG16 model is employed in a variety of deep learning image classification challenges, smaller network topologies like GoogleNet and SqueezeNet are frequently preferred. In any case, the VGGNet is an excellent learning building block because it is simple to implement.

In the ILSVRC-2012 and ILSVRC-2013 contests, VGG16 outperformed prior versions of models. Furthermore, the VGG16 result is fighting for the classification task winner (GoogLeNet with 6.7 percent error) and exceeds the ILSVRC-2013 winning submission Clarifai by a significant margin. It scored roughly 11.7 percent using external training data and 11.2 percent without. In terms of single-net performance, the VGGNet-16 model outperforms a single GoogLeNet by roughly 0.9 percent, with about 7.0 percent test error.

Advantages of VGG16:

As the number of layers in CNN grows, the model's ability to fit more complex functions grows as well. As a result, more layers imply better performance. This is not the same as an Artificial Neural Network (ANN), where increasing the number of layers does not always result in higher performance.

In both the ILSVRC-2012 and ILSVRC-2013 competitions, VGG16 outperformed the previous generation of models. In terms of single-net performance, the VGG16 architecture came out on top (7.0 percent test error). The mistake rates are listed in the table below.

When working with a VGG network, there are two major downsides to be mindful of. To begin with, training requires time. Second, the weights associated with network architecture are substantial. The trained VGG16 model is almost 500MB in size because to its depth and amount of completely linked nodes. Smaller network topologies are often preferred over VGG16 in many deep learning image categorization challenges (such as SqueezeNet, GoogleNet, etc.

6. SOURCE-CODE

In [2]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
import numpy as np
import matplotlib.pyplot as plt
import os
import shutil
import math
```

PATH CREATION AND CHECK

In [4]:
```
ROOT_DIR = '../input/covid19-lungs-binaryclassification/Covid19_Lungs_BinaryClassification'
no_of_images = {}

for dir in os.listdir(ROOT_DIR):
    no_of_images[dir] = len(os.listdir(os.path.join(ROOT_DIR,dir)))

no_of_images.items()
```

Out[4]:
```
dict_items([('Negative', 8446), ('Positive', 846)])
```

SPLIT DATA

70% TRAIN 15% VAL 15% TEST

In [5]:
```
os.mkdir("./Data")
def dataFolder(p, split):
    if not os.path.exists("./Data/"+p):
        os.mkdir("./Data/"+p)
        for dir in os.listdir(ROOT_DIR):
            os.makedirs("./Data/"+p+"/"+dir)
            for img in np.random.choice(a=os.listdir(os.path.join(ROOT_DIR,dir)), size = (math.floor(split*no_of_images[dir])
-5),replace=False):
                O = os.path.join(ROOT_DIR,dir,img)
                D = os.path.join("./Data/"+p,dir)
                shutil.copy(O,D)
    else:
        print(f"{p} Exists")
```

In [9]:
```
dataFolder("Train",0.7)
```

```
Train Exists
```

In [10]:
```
dataFolder("Val",0.15)
```

```
Val Exists
```

In [11]:
```
dataFolder("Test",0.15)
```

```
Test Exists
```

MODEL CREATION

In [21]:

```
from keras.models import Model
from keras.applications.inception_v3 import InceptionV3

from keras.layers import Conv2D, MaxPool2D, Flatten, Dense
import keras.losses
import keras
from keras.applications.vgg16 import VGG16
from keras.applications.densenet import DenseNet169
from keras.applications.resnet_v2 import ResNet152V2
```

DenseNet169

In [16]:

```
base_model = DenseNet169(input_shape=(256,256,3),include_top=False)
for layer in base_model.layers:
    layer.trainable=False

X=Flatten()(base_model.output)
X=Dense(units=2, activation='sigmoid')(X)

model_DN = Model(base_model.input, X)

model_DN.compile(optimizer='adam',loss=keras.losses.binary_crossentropy,metrics=['accuracy'])
model_DN.summary()
```

```
Model: "model_3"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_4 (InputLayer)            [(None, 256, 256, 3) 0
__________________________________________________________________________________________________
zero_padding2d_2 (ZeroPadding2D (None, 262, 262, 3)  0           input_4[0][0]
__________________________________________________________________________________________________
conv1/conv (Conv2D)             (None, 128, 128, 64) 9408        zero_padding2d_2[0][0]
__________________________________________________________________________________________________
conv1/bn (BatchNormalization)   (None, 128, 128, 64) 256         conv1/conv[0][0]
__________________________________________________________________________________________________
conv1/relu (Activation)         (None, 128, 128, 64) 0           conv1/bn[0][0]
__________________________________________________________________________________________________
zero_padding2d_3 (ZeroPadding2D (None, 130, 130, 64) 0           conv1/relu[0][0]
__________________________________________________________________________________________________
pool1 (MaxPooling2D)            (None, 64, 64, 64)   0           zero_padding2d_3[0][0]
__________________________________________________________________________________________________
conv2_block1_0_bn (BatchNormali (None, 64, 64, 64)   256         pool1[0][0]
__________________________________________________________________________________________________
conv2_block1_0_relu (Activation (None, 64, 64, 64)   0           conv2_block1_0_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_1_conv (Conv2D)    (None, 64, 64, 128)  8192        conv2_block1_0_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_1_bn (BatchNormali (None, 64, 64, 128)  512         conv2_block1_1_conv[0][0]
__________________________________________________________________________________________________
conv2_block1_1_relu (Activation (None, 64, 64, 128)  0           conv2_block1_1_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_2_conv (Conv2D)    (None, 64, 64, 32)   36864       conv2_block1_1_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_concat (Concatenat (None, 64, 64, 96)   0           pool1[0][0]
                                                                 conv2_block1_2_conv[0][0]
__________________________________________________________________________________________________
conv2_block2_0_bn (BatchNormali (None, 64, 64, 96)   384         conv2_block1_concat[0][0]
__________________________________________________________________________________________________
conv2_block2_0_relu (Activation (None, 64, 64, 96)   0           conv2_block2_0_bn[0][0]
__________________________________________________________________________________________________
conv2_block2_1_conv (Conv2D)    (None, 64, 64, 128)  12288       conv2_block2_0_relu[0][0]
__________________________________________________________________________________________________
conv2_block2_1_bn (BatchNormali (None, 64, 64, 128)  512         conv2_block2_1_conv[0][0]
```

conv2_block2_1_relu (Activation	(None, 64, 64, 128)	0	conv2_block2_1_bn[0][0]
conv2_block2_2_conv (Conv2D)	(None, 64, 64, 32)	36864	conv2_block2_1_relu[0][0]
conv2_block2_concat (Concatenat	(None, 64, 64, 128)	0	conv2_block1_concat[0][0] conv2_block2_2_conv[0][0]
conv2_block3_0_bn (BatchNormali	(None, 64, 64, 128)	512	conv2_block2_concat[0][0]
conv2_block3_0_relu (Activation	(None, 64, 64, 128)	0	conv2_block3_0_bn[0][0]
conv2_block3_1_conv (Conv2D)	(None, 64, 64, 128)	16384	conv2_block3_0_relu[0][0]
conv2_block3_1_bn (BatchNormali	(None, 64, 64, 128)	512	conv2_block3_1_conv[0][0]
conv2_block3_1_relu (Activation	(None, 64, 64, 128)	0	conv2_block3_1_bn[0][0]
conv2_block3_2_conv (Conv2D)	(None, 64, 64, 32)	36864	conv2_block3_1_relu[0][0]
conv2_block3_concat (Concatenat	(None, 64, 64, 160)	0	conv2_block2_concat[0][0] conv2_block3_2_conv[0][0]
conv2_block4_0_bn (BatchNormali	(None, 64, 64, 160)	640	conv2_block3_concat[0][0]
conv2_block4_0_relu (Activation	(None, 64, 64, 160)	0	conv2_block4_0_bn[0][0]
conv2_block4_1_conv (Conv2D)	(None, 64, 64, 128)	20480	conv2_block4_0_relu[0][0]
conv2_block4_1_bn (BatchNormali	(None, 64, 64, 128)	512	conv2_block4_1_conv[0][0]
conv2_block4_1_relu (Activation	(None, 64, 64, 128)	0	conv2_block4_1_bn[0][0]
conv2_block4_2_conv (Conv2D)	(None, 64, 64, 32)	36864	conv2_block4_1_relu[0][0]
conv2_block4_concat (Concatenat	(None, 64, 64, 192)	0	conv2_block3_concat[0][0] conv2_block4_2_conv[0][0]
conv2_block5_0_bn (BatchNormali	(None, 64, 64, 192)	768	conv2_block4_concat[0][0]
conv2_block5_0_relu (Activation	(None, 64, 64, 192)	0	conv2_block5_0_bn[0][0]
conv2_block5_1_conv (Conv2D)	(None, 64, 64, 128)	24576	conv2_block5_0_relu[0][0]
conv2_block5_1_bn (BatchNormali	(None, 64, 64, 128)	512	conv2_block5_1_conv[0][0]
conv2_block5_1_relu (Activation	(None, 64, 64, 128)	0	conv2_block5_1_bn[0][0]
conv2_block5_2_conv (Conv2D)	(None, 64, 64, 32)	36864	conv2_block5_1_relu[0][0]
conv2_block5_concat (Concatenat	(None, 64, 64, 224)	0	conv2_block4_concat[0][0] conv2_block5_2_conv[0][0]
conv2_block6_0_bn (BatchNormali	(None, 64, 64, 224)	896	conv2_block5_concat[0][0]
conv2_block6_0_relu (Activation	(None, 64, 64, 224)	0	conv2_block6_0_bn[0][0]
conv2_block6_1_conv (Conv2D)	(None, 64, 64, 128)	28672	conv2_block6_0_relu[0][0]
conv2_block6_1_bn (BatchNormali	(None, 64, 64, 128)	512	conv2_block6_1_conv[0][0]
conv2_block6_1_relu (Activation	(None, 64, 64, 128)	0	conv2_block6_1_bn[0][0]
conv2_block6_2_conv (Conv2D)	(None, 64, 64, 32)	36864	conv2_block6_1_relu[0][0]
conv2_block6_concat (Concatenat	(None, 64, 64, 256)	0	conv2_block5_concat[0][0] conv2_block6_2_conv[0][0]
pool2_bn (BatchNormalization)	(None, 64, 64, 256)	1024	conv2_block6_concat[0][0]
pool2_relu (Activation)	(None, 64, 64, 256)	0	pool2_bn[0][0]
pool2_conv (Conv2D)	(None, 64, 64, 128)	32768	pool2_relu[0][0]
pool2_pool (AveragePooling2D)	(None, 32, 32, 128)	0	pool2_conv[0][0]
conv3_block1_0_bn (BatchNormali	(None, 32, 32, 128)	512	pool2_pool[0][0]
conv3_block1_0_relu (Activation	(None, 32, 32, 128)	0	conv3_block1_0_bn[0][0]
conv3_block1_1_conv (Conv2D)	(None, 32, 32, 128)	16384	conv3_block1_0_relu[0][0]
conv3_block1_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block1_1_conv[0][0]
conv3_block1_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block1_1_bn[0][0]
conv3_block1_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block1_1_relu[0][0]
conv3_block1_concat (Concatenat	(None, 32, 32, 160)	0	pool2_pool[0][0] conv3_block1_2_conv[0][0]
conv3_block2_0_bn (BatchNormali	(None, 32, 32, 160)	640	conv3_block1_concat[0][0]
conv3_block2_0_relu (Activation	(None, 32, 32, 160)	0	conv3_block2_0_bn[0][0]
conv3_block2_1_conv (Conv2D)	(None, 32, 32, 128)	20480	conv3_block2_0_relu[0][0]
conv3_block2_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block2_1_conv[0][0]
conv3_block2_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block2_1_bn[0][0]
conv3_block2_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block2_1_relu[0][0]
conv3_block2_concat (Concatenat	(None, 32, 32, 192)	0	conv3_block1_concat[0][0] conv3_block2_2_conv[0][0]
conv3_block3_0_bn (BatchNormali	(None, 32, 32, 192)	768	conv3_block2_concat[0][0]
conv3_block3_0_relu (Activation	(None, 32, 32, 192)	0	conv3_block3_0_bn[0][0]
conv3_block3_1_conv (Conv2D)	(None, 32, 32, 128)	24576	conv3_block3_0_relu[0][0]
conv3_block3_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block3_1_conv[0][0]
conv3_block3_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block3_1_bn[0][0]
conv3_block3_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block3_1_relu[0][0]
conv3_block3_concat (Concatenat	(None, 32, 32, 224)	0	conv3_block2_concat[0][0] conv3_block3_2_conv[0][0]
conv3_block4_0_bn (BatchNormali	(None, 32, 32, 224)	896	conv3_block3_concat[0][0]
conv3_block4_0_relu (Activation	(None, 32, 32, 224)	0	conv3_block4_0_bn[0][0]
conv3_block4_1_conv (Conv2D)	(None, 32, 32, 128)	28672	conv3_block4_0_relu[0][0]
conv3_block4_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block4_1_conv[0][0]
conv3_block4_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block4_1_bn[0][0]
conv3_block4_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block4_1_relu[0][0]
conv3_block4_concat (Concatenat	(None, 32, 32, 256)	0	conv3_block3_concat[0][0] conv3_block4_2_conv[0][0]
conv3_block5_0_bn (BatchNormali	(None, 32, 32, 256)	1024	conv3_block4_concat[0][0]
conv3_block5_0_relu (Activation	(None, 32, 32, 256)	0	conv3_block5_0_bn[0][0]
conv3_block5_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block5_1_conv[0][0]
conv3_block5_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block5_1_bn[0][0]
conv3_block5_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block5_1_relu[0][0]
conv3_block5_concat (Concatenat	(None, 32, 32, 288)	0	conv3_block4_concat[0][0] conv3_block5_2_conv[0][0]
conv3_block6_0_bn (BatchNormali	(None, 32, 32, 288)	1152	conv3_block5_concat[0][0]
conv3_block6_0_relu (Activation	(None, 32, 32, 288)	0	conv3_block6_0_bn[0][0]
conv3_block6_1_conv (Conv2D)	(None, 32, 32, 128)	36864	conv3_block6_0_relu[0][0]
conv3_block6_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block6_1_conv[0][0]
conv3_block6_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block6_1_bn[0][0]
conv3_block6_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block6_1_relu[0][0]
conv3_block6_concat (Concatenat	(None, 32, 32, 320)	0	conv3_block5_concat[0][0] conv3_block6_2_conv[0][0]
conv3_block7_0_bn (BatchNormali	(None, 32, 32, 320)	1280	conv3_block6_concat[0][0]
conv3_block5_1_conv (Conv2D)	(None, 32, 32, 128)	32768	conv3_block5_0_relu[0][0]
conv3_block7_0_relu (Activation	(None, 32, 32, 320)	0	conv3_block7_0_bn[0][0]
conv3_block7_1_conv (Conv2D)	(None, 32, 32, 128)	40960	conv3_block7_0_relu[0][0]
conv3_block7_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block7_1_conv[0][0]
conv3_block7_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block7_1_bn[0][0]
conv3_block7_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block7_1_relu[0][0]
conv3_block7_concat (Concatenat	(None, 32, 32, 352)	0	conv3_block6_concat[0][0] conv3_block7_2_conv[0][0]
conv3_block8_0_bn (BatchNormali	(None, 32, 32, 352)	1408	conv3_block7_concat[0][0]
conv3_block8_0_relu (Activation	(None, 32, 32, 352)	0	conv3_block8_0_bn[0][0]
conv3_block8_1_conv (Conv2D)	(None, 32, 32, 128)	45056	conv3_block8_0_relu[0][0]
conv3_block8_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block8_1_conv[0][0]
conv3_block8_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block8_1_bn[0][0]
conv3_block8_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block8_1_relu[0][0]
conv3_block8_concat (Concatenat	(None, 32, 32, 384)	0	conv3_block7_concat[0][0] conv3_block8_2_conv[0][0]
conv3_block9_0_bn (BatchNormali	(None, 32, 32, 384)	1536	conv3_block8_concat[0][0]
conv3_block9_0_relu (Activation	(None, 32, 32, 384)	0	conv3_block9_0_bn[0][0]
conv3_block9_1_conv (Conv2D)	(None, 32, 32, 128)	49152	conv3_block9_0_relu[0][0]
conv3_block9_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block9_1_conv[0][0]
conv3_block9_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block9_1_bn[0][0]
conv3_block9_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block9_1_relu[0][0]
conv3_block9_concat (Concatenat	(None, 32, 32, 416)	0	conv3_block8_concat[0][0] conv3_block9_2_conv[0][0]
conv3_block10_0_bn (BatchNormal	(None, 32, 32, 416)	1664	conv3_block9_concat[0][0]
conv3_block10_0_relu (Activatio	(None, 32, 32, 416)	0	conv3_block10_0_bn[0][0]
conv3_block10_1_conv (Conv2D)	(None, 32, 32, 128)	53248	conv3_block10_0_relu[0][0]
conv3_block10_1_bn (BatchNormal	(None, 32, 32, 128)	512	conv3_block10_1_conv[0][0]
conv3_block10_1_relu (Activatio	(None, 32, 32, 128)	0	conv3_block10_1_bn[0][0]
conv3_block10_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block10_1_relu[0][0]
conv3_block10_concat (Concatena	(None, 32, 32, 448)	0	conv3_block9_concat[0][0] conv3_block10_2_conv[0][0]
conv3_block11_0_bn (BatchNormal	(None, 32, 32, 448)	1792	conv3_block10_concat[0][0]
conv3_block11_0_relu (Activatio	(None, 32, 32, 448)	0	conv3_block11_0_bn[0][0]
conv3_block11_1_conv (Conv2D)	(None, 32, 32, 128)	57344	conv3_block11_0_relu[0][0]
conv3_block11_1_bn (BatchNormal	(None, 32, 32, 128)	512	conv3_block11_1_conv[0][0]
conv3_block11_1_relu (Activatio	(None, 32, 32, 128)	0	conv3_block11_1_bn[0][0]
conv3_block11_2_conv (Conv2D)	(None, 32, 32, 32)	36864	conv3_block11_1_relu[0][0]

```
conv3_block11_concat (Concatena (None, 32, 32, 480)  0           conv3_block10_concat[0][0]
                                                                  conv3_block11_2_conv[0][0]
__________________________________________________________________________________________________
conv3_block12_0_bn (BatchNormal (None, 32, 32, 480)  1920        conv3_block11_concat[0][0]
__________________________________________________________________________________________________
conv3_block12_0_relu (Activatio (None, 32, 32, 480)  0           conv3_block12_0_bn[0][0]
__________________________________________________________________________________________________
conv3_block12_1_conv (Conv2D)   (None, 32, 32, 128)  61440       conv3_block12_0_relu[0][0]
__________________________________________________________________________________________________
conv3_block12_1_bn (BatchNormal (None, 32, 32, 128)  512         conv3_block12_1_conv[0][0]
__________________________________________________________________________________________________
conv3_block12_1_relu (Activatio (None, 32, 32, 128)  0           conv3_block12_1_bn[0][0]
__________________________________________________________________________________________________
conv3_block12_2_conv (Conv2D)   (None, 32, 32, 32)   36864       conv3_block12_1_relu[0][0]
__________________________________________________________________________________________________
conv3_block12_concat (Concatena (None, 32, 32, 512)  0           conv3_block11_concat[0][0]
                                                                  conv3_block12_2_conv[0][0]
__________________________________________________________________________________________________
pool3_bn (BatchNormalization)   (None, 32, 32, 512)  2048        conv3_block12_concat[0][0]
__________________________________________________________________________________________________
pool3_relu (Activation)         (None, 32, 32, 512)  0           pool3_bn[0][0]
__________________________________________________________________________________________________
pool3_conv (Conv2D)             (None, 32, 32, 256)  131072      pool3_relu[0][0]
__________________________________________________________________________________________________
pool3_pool (AveragePooling2D)   (None, 16, 16, 256)  0           pool3_conv[0][0]
__________________________________________________________________________________________________
__________________________________________________________________________________________________
conv4_block1_0_bn (BatchNormali (None, 16, 16, 256)  1024        pool3_pool[0][0]
__________________________________________________________________________________________________
conv4_block1_0_relu (Activation (None, 16, 16, 256)  0           conv4_block1_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block1_1_conv (Conv2D)    (None, 16, 16, 128)  32768       conv4_block1_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block1_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block1_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block1_1_relu (Activation (None, 16, 16, 128)  0           conv4_block1_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block1_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block1_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block1_concat (Concatenat (None, 16, 16, 288)  0           pool3_pool[0][0]
                                                                  conv4_block1_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block2_0_bn (BatchNormali (None, 16, 16, 288)  1152        conv4_block1_concat[0][0]
__________________________________________________________________________________________________
conv4_block2_0_relu (Activation (None, 16, 16, 288)  0           conv4_block2_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block2_1_conv (Conv2D)    (None, 16, 16, 128)  36864       conv4_block2_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block2_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block2_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block2_1_relu (Activation (None, 16, 16, 128)  0           conv4_block2_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block2_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block2_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block2_concat (Concatenat (None, 16, 16, 320)  0           conv4_block1_concat[0][0]
                                                                  conv4_block2_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block3_0_bn (BatchNormali (None, 16, 16, 320)  1280        conv4_block2_concat[0][0]
__________________________________________________________________________________________________
conv4_block3_0_relu (Activation (None, 16, 16, 320)  0           conv4_block3_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block3_1_conv (Conv2D)    (None, 16, 16, 128)  40960       conv4_block3_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block3_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block3_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block3_1_relu (Activation (None, 16, 16, 128)  0           conv4_block3_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block3_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block3_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block3_concat (Concatenat (None, 16, 16, 352)  0           conv4_block2_concat[0][0]
                                                                  conv4_block3_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block4_0_bn (BatchNormali (None, 16, 16, 352)  1408        conv4_block3_concat[0][0]
__________________________________________________________________________________________________
conv4_block4_0_relu (Activation (None, 16, 16, 352)  0           conv4_block4_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block4_1_conv (Conv2D)    (None, 16, 16, 128)  45056       conv4_block4_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block4_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block4_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block4_1_relu (Activation (None, 16, 16, 128)  0           conv4_block4_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block4_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block4_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block4_concat (Concatenat (None, 16, 16, 384)  0           conv4_block3_concat[0][0]
                                                                  conv4_block4_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block5_0_bn (BatchNormali (None, 16, 16, 384)  1536        conv4_block4_concat[0][0]
__________________________________________________________________________________________________
conv4_block5_0_relu (Activation (None, 16, 16, 384)  0           conv4_block5_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block5_1_conv (Conv2D)    (None, 16, 16, 128)  49152       conv4_block5_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block5_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block5_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block5_1_relu (Activation (None, 16, 16, 128)  0           conv4_block5_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block5_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block5_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block5_concat (Concatenat (None, 16, 16, 416)  0           conv4_block4_concat[0][0]
                                                                  conv4_block5_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block6_0_bn (BatchNormali (None, 16, 16, 416)  1664        conv4_block5_concat[0][0]
__________________________________________________________________________________________________
conv4_block6_0_relu (Activation (None, 16, 16, 416)  0           conv4_block6_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block6_1_conv (Conv2D)    (None, 16, 16, 128)  53248       conv4_block6_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block6_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block6_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block6_1_relu (Activation (None, 16, 16, 128)  0           conv4_block6_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block6_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block6_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block6_concat (Concatenat (None, 16, 16, 448)  0           conv4_block5_concat[0][0]
                                                                  conv4_block6_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block7_0_bn (BatchNormali (None, 16, 16, 448)  1792        conv4_block6_concat[0][0]
__________________________________________________________________________________________________
conv4_block7_0_relu (Activation (None, 16, 16, 448)  0           conv4_block7_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block7_1_conv (Conv2D)    (None, 16, 16, 128)  57344       conv4_block7_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block7_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block7_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block7_1_relu (Activation (None, 16, 16, 128)  0           conv4_block7_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block7_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block7_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block7_concat (Concatenat (None, 16, 16, 480)  0           conv4_block6_concat[0][0]
                                                                  conv4_block7_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block8_0_bn (BatchNormali (None, 16, 16, 480)  1920        conv4_block7_concat[0][0]
__________________________________________________________________________________________________
conv4_block8_0_relu (Activation (None, 16, 16, 480)  0           conv4_block8_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block8_1_conv (Conv2D)    (None, 16, 16, 128)  61440       conv4_block8_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block8_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block8_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block8_1_relu (Activation (None, 16, 16, 128)  0           conv4_block8_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block8_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block8_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block8_concat (Concatenat (None, 16, 16, 512)  0           conv4_block7_concat[0][0]
                                                                  conv4_block8_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block9_0_bn (BatchNormali (None, 16, 16, 512)  2048        conv4_block8_concat[0][0]
__________________________________________________________________________________________________
conv4_block9_0_relu (Activation (None, 16, 16, 512)  0           conv4_block9_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block9_1_conv (Conv2D)    (None, 16, 16, 128)  65536       conv4_block9_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block9_1_bn (BatchNormali (None, 16, 16, 128)  512         conv4_block9_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block9_1_relu (Activation (None, 16, 16, 128)  0           conv4_block9_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block9_2_conv (Conv2D)    (None, 16, 16, 32)   36864       conv4_block9_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block9_concat (Concatenat (None, 16, 16, 544)  0           conv4_block8_concat[0][0]
                                                                  conv4_block9_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block10_0_bn (BatchNormal (None, 16, 16, 544)  2176        conv4_block9_concat[0][0]
__________________________________________________________________________________________________
conv4_block10_0_relu (Activatio (None, 16, 16, 544)  0           conv4_block10_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block10_1_conv (Conv2D)   (None, 16, 16, 128)  69632       conv4_block10_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block10_1_bn (BatchNormal (None, 16, 16, 128)  512         conv4_block10_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block10_1_relu (Activatio (None, 16, 16, 128)  0           conv4_block10_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block10_2_conv (Conv2D)   (None, 16, 16, 32)   36864       conv4_block10_1_relu[0][0]
__________________________________________________________________________________________________
conv4_block10_concat (Concatena (None, 16, 16, 576)  0           conv4_block9_concat[0][0]
                                                                  conv4_block10_2_conv[0][0]
__________________________________________________________________________________________________
conv4_block11_0_bn (BatchNormal (None, 16, 16, 576)  2304        conv4_block10_concat[0][0]
__________________________________________________________________________________________________
conv4_block11_0_relu (Activatio (None, 16, 16, 576)  0           conv4_block11_0_bn[0][0]
__________________________________________________________________________________________________
conv4_block11_1_conv (Conv2D)   (None, 16, 16, 128)  73728       conv4_block11_0_relu[0][0]
__________________________________________________________________________________________________
conv4_block11_1_bn (BatchNormal (None, 16, 16, 128)  512         conv4_block11_1_conv[0][0]
__________________________________________________________________________________________________
conv4_block11_1_relu (Activatio (None, 16, 16, 128)  0           conv4_block11_1_bn[0][0]
__________________________________________________________________________________________________
conv4_block11_2_conv (Conv2D)   (None, 16, 16, 32)   36864       conv4_block11_1_relu[0][0]
__________________________________________________________________________________________________
```

conv4_block11_concat (Concatena	(None, 16, 16, 608)	0	conv4_block10_concat[0][0] conv4_block11_2_conv[0][0]
conv4_block12_0_bn (BatchNormal	(None, 16, 16, 608)	2432	conv4_block11_concat[0][0]
conv4_block12_0_relu (Activatio	(None, 16, 16, 608)	0	conv4_block12_0_bn[0][0]
conv4_block12_1_conv (Conv2D)	(None, 16, 16, 128)	77824	conv4_block12_0_relu[0][0]
conv4_block12_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block12_1_conv[0][0]
conv4_block12_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block12_1_bn[0][0]
conv4_block12_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block12_1_relu[0][0]
conv4_block12_concat (Concatena	(None, 16, 16, 640)	0	conv4_block11_concat[0][0] conv4_block12_2_conv[0][0]
conv4_block13_0_bn (BatchNormal	(None, 16, 16, 640)	2560	conv4_block12_concat[0][0]
conv4_block13_0_relu (Activatio	(None, 16, 16, 640)	0	conv4_block13_0_bn[0][0]
conv4_block13_1_conv (Conv2D)	(None, 16, 16, 128)	81920	conv4_block13_0_relu[0][0]
conv4_block13_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block13_1_conv[0][0]
conv4_block13_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block13_1_bn[0][0]
conv4_block13_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block13_1_relu[0][0]
conv4_block13_concat (Concatena	(None, 16, 16, 672)	0	conv4_block12_concat[0][0] conv4_block13_2_conv[0][0]
conv4_block14_0_bn (BatchNormal	(None, 16, 16, 672)	2688	conv4_block13_concat[0][0]
conv4_block14_0_relu (Activatio	(None, 16, 16, 672)	0	conv4_block14_0_bn[0][0]
conv4_block14_1_conv (Conv2D)	(None, 16, 16, 128)	86016	conv4_block14_0_relu[0][0]
conv4_block14_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block14_1_conv[0][0]
conv4_block14_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block14_1_bn[0][0]
conv4_block14_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block14_1_relu[0][0]
conv4_block14_concat (Concatena	(None, 16, 16, 704)	0	conv4_block13_concat[0][0] conv4_block14_2_conv[0][0]
conv4_block15_0_bn (BatchNormal	(None, 16, 16, 704)	2816	conv4_block14_concat[0][0]
conv4_block15_0_relu (Activatio	(None, 16, 16, 704)	0	conv4_block15_0_bn[0][0]
conv4_block15_1_conv (Conv2D)	(None, 16, 16, 128)	90112	conv4_block15_0_relu[0][0]
conv4_block15_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block15_1_conv[0][0]
conv4_block15_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block15_1_bn[0][0]
conv4_block15_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block15_1_relu[0][0]
conv4_block15_concat (Concatena	(None, 16, 16, 736)	0	conv4_block14_concat[0][0] conv4_block15_2_conv[0][0]
conv4_block16_0_bn (BatchNormal	(None, 16, 16, 736)	2944	conv4_block15_concat[0][0]
conv4_block16_0_relu (Activatio	(None, 16, 16, 736)	0	conv4_block16_0_bn[0][0]
conv4_block16_1_conv (Conv2D)	(None, 16, 16, 128)	94208	conv4_block16_0_relu[0][0]
conv4_block16_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block16_1_conv[0][0]
conv4_block16_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block16_1_bn[0][0]
conv4_block16_1_conv (Conv2D)	(None, 16, 16, 128)	94208	conv4_block16_0_relu[0][0]
conv4_block16_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block16_1_conv[0][0]
conv4_block16_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block16_1_bn[0][0]
conv4_block16_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block16_1_relu[0][0]
conv4_block16_concat (Concatena	(None, 16, 16, 768)	0	conv4_block15_concat[0][0] conv4_block16_2_conv[0][0]
conv4_block17_0_bn (BatchNormal	(None, 16, 16, 768)	3072	conv4_block16_concat[0][0]
conv4_block17_0_relu (Activatio	(None, 16, 16, 768)	0	conv4_block17_0_bn[0][0]
conv4_block17_1_conv (Conv2D)	(None, 16, 16, 128)	98304	conv4_block17_0_relu[0][0]
conv4_block17_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block17_1_conv[0][0]
conv4_block17_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block17_1_bn[0][0]
conv4_block17_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block17_1_relu[0][0]
conv4_block17_concat (Concatena	(None, 16, 16, 800)	0	conv4_block16_concat[0][0] conv4_block17_2_conv[0][0]
conv4_block18_0_bn (BatchNormal	(None, 16, 16, 800)	3200	conv4_block17_concat[0][0]
conv4_block18_0_relu (Activatio	(None, 16, 16, 800)	0	conv4_block18_0_bn[0][0]
conv4_block18_1_conv (Conv2D)	(None, 16, 16, 128)	102400	conv4_block18_0_relu[0][0]
conv4_block18_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block18_1_conv[0][0]
conv4_block18_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block18_1_bn[0][0]
conv4_block18_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block18_1_relu[0][0]
conv4_block18_concat (Concatena	(None, 16, 16, 832)	0	conv4_block17_concat[0][0] conv4_block18_2_conv[0][0]
conv4_block19_0_bn (BatchNormal	(None, 16, 16, 832)	3328	conv4_block18_concat[0][0]
conv4_block19_0_relu (Activatio	(None, 16, 16, 832)	0	conv4_block19_0_bn[0][0]
conv4_block19_1_conv (Conv2D)	(None, 16, 16, 128)	106496	conv4_block19_0_relu[0][0]
conv4_block19_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block19_1_conv[0][0]
conv4_block19_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block19_1_bn[0][0]
conv4_block19_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block19_1_relu[0][0]
conv4_block19_concat (Concatena	(None, 16, 16, 864)	0	conv4_block18_concat[0][0] conv4_block19_2_conv[0][0]
conv4_block20_0_bn (BatchNormal	(None, 16, 16, 864)	3456	conv4_block19_concat[0][0]
conv4_block20_0_relu (Activatio	(None, 16, 16, 864)	0	conv4_block20_0_bn[0][0]
conv4_block20_1_conv (Conv2D)	(None, 16, 16, 128)	110592	conv4_block20_0_relu[0][0]
conv4_block20_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block20_1_conv[0][0]
conv4_block20_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block20_1_bn[0][0]
conv4_block20_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block20_1_relu[0][0]
conv4_block20_concat (Concatena	(None, 16, 16, 896)	0	conv4_block19_concat[0][0] conv4_block20_2_conv[0][0]
conv4_block21_0_bn (BatchNormal	(None, 16, 16, 896)	3584	conv4_block20_concat[0][0]
conv4_block21_0_relu (Activatio	(None, 16, 16, 896)	0	conv4_block21_0_bn[0][0]
conv4_block21_1_conv (Conv2D)	(None, 16, 16, 128)	114688	conv4_block21_0_relu[0][0]
conv4_block21_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block21_1_conv[0][0]
conv4_block21_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block21_1_bn[0][0]
conv4_block21_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block21_1_relu[0][0]
conv4_block21_concat (Concatena	(None, 16, 16, 928)	0	conv4_block20_concat[0][0] conv4_block21_2_conv[0][0]
conv4_block22_0_bn (BatchNormal	(None, 16, 16, 928)	3712	conv4_block21_concat[0][0]
conv4_block22_0_relu (Activatio	(None, 16, 16, 928)	0	conv4_block22_0_bn[0][0]
conv4_block22_1_conv (Conv2D)	(None, 16, 16, 128)	118784	conv4_block22_0_relu[0][0]
conv4_block22_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block22_1_conv[0][0]
conv4_block22_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block22_1_bn[0][0]
conv4_block22_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block22_1_relu[0][0]
conv4_block22_concat (Concatena	(None, 16, 16, 960)	0	conv4_block21_concat[0][0] conv4_block22_2_conv[0][0]
conv4_block23_0_bn (BatchNormal	(None, 16, 16, 960)	3840	conv4_block22_concat[0][0]
conv4_block23_0_relu (Activatio	(None, 16, 16, 960)	0	conv4_block23_0_bn[0][0]
conv4_block23_1_conv (Conv2D)	(None, 16, 16, 128)	122880	conv4_block23_0_relu[0][0]
conv4_block23_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block23_1_conv[0][0]
conv4_block23_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block23_1_bn[0][0]
conv4_block23_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block23_1_relu[0][0]
conv4_block23_concat (Concatena	(None, 16, 16, 992)	0	conv4_block22_concat[0][0] conv4_block23_2_conv[0][0]
conv4_block24_0_bn (BatchNormal	(None, 16, 16, 992)	3968	conv4_block23_concat[0][0]
conv4_block24_0_relu (Activatio	(None, 16, 16, 992)	0	conv4_block24_0_bn[0][0]
conv4_block24_1_conv (Conv2D)	(None, 16, 16, 128)	126976	conv4_block24_0_relu[0][0]
conv4_block24_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block24_1_conv[0][0]

conv4_block24_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block24_1_bn[0][0]
conv4_block24_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block24_1_relu[0][0]
conv4_block24_concat (Concatena	(None, 16, 16, 1024)	0	conv4_block23_concat[0][0] conv4_block24_2_conv[0][0]
conv4_block25_0_bn (BatchNormal	(None, 16, 16, 1024)	4096	conv4_block24_concat[0][0]
conv4_block25_0_relu (Activatio	(None, 16, 16, 1024)	0	conv4_block25_0_bn[0][0]
conv4_block25_1_conv (Conv2D)	(None, 16, 16, 128)	131072	conv4_block25_0_relu[0][0]
conv4_block25_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block25_1_conv[0][0]
conv4_block25_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block25_1_bn[0][0]
conv4_block25_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block25_1_relu[0][0]
conv4_block25_concat (Concatena	(None, 16, 16, 1056)	0	conv4_block24_concat[0][0] conv4_block25_2_conv[0][0]
conv4_block26_0_bn (BatchNormal	(None, 16, 16, 1056)	4224	conv4_block25_concat[0][0]
conv4_block26_0_relu (Activatio	(None, 16, 16, 1056)	0	conv4_block26_0_bn[0][0]
conv4_block26_1_conv (Conv2D)	(None, 16, 16, 128)	135168	conv4_block26_0_relu[0][0]
conv4_block26_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block26_1_conv[0][0]
conv4_block26_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block26_1_bn[0][0]
conv4_block26_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block26_1_relu[0][0]
conv4_block26_concat (Concatena	(None, 16, 16, 1088)	0	conv4_block25_concat[0][0] conv4_block26_2_conv[0][0]
conv4_block27_0_bn (BatchNormal	(None, 16, 16, 1088)	4352	conv4_block26_concat[0][0]
conv4_block27_0_relu (Activatio	(None, 16, 16, 1088)	0	conv4_block27_0_bn[0][0]
conv4_block27_1_conv (Conv2D)	(None, 16, 16, 128)	139264	conv4_block27_0_relu[0][0]
conv4_block27_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block27_1_conv[0][0]
conv4_block27_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block27_1_bn[0][0]
conv4_block27_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block27_1_relu[0][0]
conv4_block27_concat (Concatena	(None, 16, 16, 1120)	0	conv4_block26_concat[0][0] conv4_block27_2_conv[0][0]
conv4_block28_0_bn (BatchNormal	(None, 16, 16, 1120)	4480	conv4_block27_concat[0][0]
conv4_block28_0_relu (Activatio	(None, 16, 16, 1120)	0	conv4_block28_0_bn[0][0]
conv4_block28_1_conv (Conv2D)	(None, 16, 16, 128)	143360	conv4_block28_0_relu[0][0]
conv4_block28_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block28_1_conv[0][0]
conv4_block28_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block28_1_bn[0][0]
conv4_block28_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block28_1_relu[0][0]
conv4_block28_concat (Concatena	(None, 16, 16, 1152)	0	conv4_block27_concat[0][0] conv4_block28_2_conv[0][0]
conv4_block29_0_bn (BatchNormal	(None, 16, 16, 1152)	4608	conv4_block28_concat[0][0]
conv4_block29_0_relu (Activatio	(None, 16, 16, 1152)	0	conv4_block29_0_bn[0][0]
conv4_block29_1_conv (Conv2D)	(None, 16, 16, 128)	147456	conv4_block29_0_relu[0][0]
conv4_block29_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block29_1_conv[0][0]
conv4_block29_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block29_1_bn[0][0]
conv4_block29_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block29_1_relu[0][0]
conv4_block29_concat (Concatena	(None, 16, 16, 1184)	0	conv4_block28_concat[0][0] conv4_block29_2_conv[0][0]
conv4_block30_0_bn (BatchNormal	(None, 16, 16, 1184)	4736	conv4_block29_concat[0][0]
conv4_block30_0_relu (Activatio	(None, 16, 16, 1184)	0	conv4_block30_0_bn[0][0]
conv4_block30_1_conv (Conv2D)	(None, 16, 16, 128)	151552	conv4_block30_0_relu[0][0]
conv4_block30_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block30_1_conv[0][0]
conv5_block2_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block2_1_bn[0][0]
conv5_block2_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block2_1_relu[0][0]
conv5_block6_1_conv (Conv2D)	(None, 8, 8, 128)	102400	conv5_block6_0_relu[0][0]
conv5_block6_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block6_1_conv[0][0]

conv4_block30_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block30_1_bn[0][0]
conv4_block30_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block30_1_relu[0][0]
conv4_block30_concat (Concatena	(None, 16, 16, 1216)	0	conv4_block29_concat[0][0] conv4_block30_2_conv[0][0]
conv4_block31_0_bn (BatchNormal	(None, 16, 16, 1216)	4864	conv4_block30_concat[0][0]
conv4_block31_0_relu (Activatio	(None, 16, 16, 1216)	0	conv4_block31_0_bn[0][0]
conv4_block31_1_conv (Conv2D)	(None, 16, 16, 128)	155648	conv4_block31_0_relu[0][0]
conv4_block31_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block31_1_conv[0][0]
conv4_block31_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block31_1_bn[0][0]
conv4_block31_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block31_1_relu[0][0]
conv4_block31_concat (Concatena	(None, 16, 16, 1248)	0	conv4_block30_concat[0][0] conv4_block31_2_conv[0][0]
conv4_block32_0_bn (BatchNormal	(None, 16, 16, 1248)	4992	conv4_block31_concat[0][0]
conv4_block32_0_relu (Activatio	(None, 16, 16, 1248)	0	conv4_block32_0_bn[0][0]
conv4_block32_1_conv (Conv2D)	(None, 16, 16, 128)	159744	conv4_block32_0_relu[0][0]
conv4_block32_1_bn (BatchNormal	(None, 16, 16, 128)	512	conv4_block32_1_conv[0][0]
conv4_block32_1_relu (Activatio	(None, 16, 16, 128)	0	conv4_block32_1_bn[0][0]
conv4_block32_2_conv (Conv2D)	(None, 16, 16, 32)	36864	conv4_block32_1_relu[0][0]
conv4_block32_concat (Concatena	(None, 16, 16, 1280)	0	conv4_block31_concat[0][0] conv4_block32_2_conv[0][0]
pool4_bn (BatchNormalization)	(None, 16, 16, 1280)	5120	conv4_block32_concat[0][0]
pool4_relu (Activation)	(None, 16, 16, 1280)	0	pool4_bn[0][0]
pool4_conv (Conv2D)	(None, 16, 16, 640)	819200	pool4_relu[0][0]
pool4_pool (AveragePooling2D)	(None, 8, 8, 640)	0	pool4_conv[0][0]
conv5_block1_0_bn (BatchNormali	(None, 8, 8, 640)	2560	pool4_pool[0][0]
conv5_block1_0_relu (Activation	(None, 8, 8, 640)	0	conv5_block1_0_bn[0][0]
conv5_block1_1_conv (Conv2D)	(None, 8, 8, 128)	81920	conv5_block1_0_relu[0][0]
conv5_block1_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block1_1_conv[0][0]
conv5_block1_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block1_1_bn[0][0]
conv5_block1_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block1_1_relu[0][0]
conv5_block1_concat (Concatenat	(None, 8, 8, 672)	0	pool4_pool[0][0] conv5_block1_2_conv[0][0]
conv5_block2_0_bn (BatchNormali	(None, 8, 8, 672)	2688	conv5_block1_concat[0][0]
conv5_block2_0_relu (Activation	(None, 8, 8, 672)	0	conv5_block2_0_bn[0][0]
conv5_block2_1_conv (Conv2D)	(None, 8, 8, 128)	86016	conv5_block2_0_relu[0][0]
conv5_block2_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block2_1_conv[0][0]
conv5_block2_concat (Concatenat	(None, 8, 8, 704)	0	conv5_block1_concat[0][0] conv5_block2_2_conv[0][0]
conv5_block3_0_bn (BatchNormali	(None, 8, 8, 704)	2816	conv5_block2_concat[0][0]
conv5_block3_0_relu (Activation	(None, 8, 8, 704)	0	conv5_block3_0_bn[0][0]
conv5_block3_1_conv (Conv2D)	(None, 8, 8, 128)	90112	conv5_block3_0_relu[0][0]
conv5_block3_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block3_1_conv[0][0]
conv5_block3_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block3_1_bn[0][0]
conv5_block3_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block3_1_relu[0][0]
conv5_block3_concat (Concatenat	(None, 8, 8, 736)	0	conv5_block2_concat[0][0] conv5_block3_2_conv[0][0]
conv5_block4_0_bn (BatchNormali	(None, 8, 8, 736)	2944	conv5_block3_concat[0][0]
conv5_block4_0_relu (Activation	(None, 8, 8, 736)	0	conv5_block4_0_bn[0][0]
conv5_block4_1_conv (Conv2D)	(None, 8, 8, 128)	94208	conv5_block4_0_relu[0][0]
conv5_block4_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block4_1_conv[0][0]
conv5_block4_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block4_1_bn[0][0]
conv5_block4_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block4_1_relu[0][0]
conv5_block4_concat (Concatenat	(None, 8, 8, 768)	0	conv5_block3_concat[0][0] conv5_block4_2_conv[0][0]

conv5_block4_0_bn (BatchNormali	(None, 8, 8, 736)	2944	conv5_block3_concat[0][0]
conv5_block4_0_relu (Activation	(None, 8, 8, 736)	0	conv5_block4_0_bn[0][0]
conv5_block4_1_conv (Conv2D)	(None, 8, 8, 128)	94208	conv5_block4_0_relu[0][0]
conv5_block4_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block4_1_conv[0][0]
conv5_block4_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block4_1_bn[0][0]
conv5_block4_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block4_1_relu[0][0]
conv5_block4_concat (Concatenat	(None, 8, 8, 768)	0	conv5_block3_concat[0][0] conv5_block4_2_conv[0][0]
conv5_block5_0_bn (BatchNormali	(None, 8, 8, 768)	3072	conv5_block4_concat[0][0]
conv5_block5_0_relu (Activation	(None, 8, 8, 768)	0	conv5_block5_0_bn[0][0]
conv5_block5_1_conv (Conv2D)	(None, 8, 8, 128)	98304	conv5_block5_0_relu[0][0]
conv5_block5_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block5_1_conv[0][0]
conv5_block5_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block5_1_bn[0][0]
conv5_block5_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block5_1_relu[0][0]
conv5_block5_concat (Concatenat	(None, 8, 8, 800)	0	conv5_block4_concat[0][0] conv5_block5_2_conv[0][0]
conv5_block6_0_bn (BatchNormali	(None, 8, 8, 800)	3200	conv5_block5_concat[0][0]
conv5_block6_0_relu (Activation	(None, 8, 8, 800)	0	conv5_block6_0_bn[0][0]
conv5_block6_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block6_1_bn[0][0]
conv5_block6_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block6_1_relu[0][0]
conv5_block6_concat (Concatenat	(None, 8, 8, 832)	0	conv5_block5_concat[0][0] conv5_block6_2_conv[0][0]
conv5_block7_0_bn (BatchNormali	(None, 8, 8, 832)	3328	conv5_block6_concat[0][0]
conv5_block7_0_relu (Activation	(None, 8, 8, 832)	0	conv5_block7_0_bn[0][0]
conv5_block7_1_conv (Conv2D)	(None, 8, 8, 128)	106496	conv5_block7_0_relu[0][0]
conv5_block7_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block7_1_conv[0][0]
conv5_block7_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block7_1_bn[0][0]
conv5_block7_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block7_1_relu[0][0]
conv5_block7_concat (Concatenat	(None, 8, 8, 864)	0	conv5_block6_concat[0][0] conv5_block7_2_conv[0][0]
conv5_block8_0_bn (BatchNormali	(None, 8, 8, 864)	3456	conv5_block7_concat[0][0]
conv5_block8_0_relu (Activation	(None, 8, 8, 864)	0	conv5_block8_0_bn[0][0]
conv5_block8_1_conv (Conv2D)	(None, 8, 8, 128)	110592	conv5_block8_0_relu[0][0]
conv5_block8_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block8_1_conv[0][0]
conv5_block8_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block8_1_bn[0][0]
conv5_block8_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block8_1_relu[0][0]
conv5_block8_concat (Concatenat	(None, 8, 8, 896)	0	conv5_block7_concat[0][0] conv5_block8_2_conv[0][0]
conv5_block9_0_bn (BatchNormali	(None, 8, 8, 896)	3584	conv5_block8_concat[0][0]
conv5_block9_0_relu (Activation	(None, 8, 8, 896)	0	conv5_block9_0_bn[0][0]
conv5_block9_1_conv (Conv2D)	(None, 8, 8, 128)	114688	conv5_block9_0_relu[0][0]
conv5_block9_1_bn (BatchNormali	(None, 8, 8, 128)	512	conv5_block9_1_conv[0][0]
conv5_block9_1_relu (Activation	(None, 8, 8, 128)	0	conv5_block9_1_bn[0][0]
conv5_block9_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block9_1_relu[0][0]
conv5_block9_concat (Concatenat	(None, 8, 8, 928)	0	conv5_block8_concat[0][0] conv5_block9_2_conv[0][0]
conv5_block10_0_bn (BatchNormal	(None, 8, 8, 928)	3712	conv5_block9_concat[0][0]
conv5_block10_0_relu (Activatio	(None, 8, 8, 928)	0	conv5_block10_0_bn[0][0]
conv5_block10_1_conv (Conv2D)	(None, 8, 8, 128)	118784	conv5_block10_0_relu[0][0]
conv5_block10_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block10_1_conv[0][0]
conv5_block10_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block10_1_bn[0][0]
conv5_block10_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block10_1_relu[0][0]
conv5_block10_concat (Concatena	(None, 8, 8, 960)	0	conv5_block9_concat[0][0] conv5_block10_2_conv[0][0]
conv5_block11_0_relu (Activatio	(None, 8, 8, 960)	0	conv5_block11_0_bn[0][0]
conv5_block11_1_conv (Conv2D)	(None, 8, 8, 128)	122880	conv5_block11_0_relu[0][0]
conv5_block11_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block11_1_conv[0][0]
conv5_block11_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block11_1_bn[0][0]
conv5_block11_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block11_1_relu[0][0]
conv5_block11_concat (Concatena	(None, 8, 8, 992)	0	conv5_block10_concat[0][0] conv5_block11_2_conv[0][0]
conv5_block12_0_bn (BatchNormal	(None, 8, 8, 992)	3968	conv5_block11_concat[0][0]
conv5_block12_0_relu (Activatio	(None, 8, 8, 992)	0	conv5_block12_0_bn[0][0]
conv5_block12_1_conv (Conv2D)	(None, 8, 8, 128)	126976	conv5_block12_0_relu[0][0]
conv5_block12_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block12_1_conv[0][0]
conv5_block12_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block12_1_bn[0][0]
conv5_block12_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block12_1_relu[0][0]
conv5_block12_concat (Concatena	(None, 8, 8, 1024)	0	conv5_block11_concat[0][0] conv5_block12_2_conv[0][0]
conv5_block13_0_bn (BatchNormal	(None, 8, 8, 1024)	4096	conv5_block12_concat[0][0]
conv5_block13_0_relu (Activatio	(None, 8, 8, 1024)	0	conv5_block13_0_bn[0][0]
conv5_block13_1_conv (Conv2D)	(None, 8, 8, 128)	131072	conv5_block13_0_relu[0][0]
conv5_block11_0_bn (BatchNormal	(None, 8, 8, 960)	3840	conv5_block10_concat[0][0]
conv5_block13_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block13_1_conv[0][0]
conv5_block13_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block13_1_bn[0][0]
conv5_block13_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block13_1_relu[0][0]
conv5_block13_concat (Concatena	(None, 8, 8, 1056)	0	conv5_block12_concat[0][0] conv5_block13_2_conv[0][0]
conv5_block14_0_bn (BatchNormal	(None, 8, 8, 1056)	4224	conv5_block13_concat[0][0]
conv5_block14_0_relu (Activatio	(None, 8, 8, 1056)	0	conv5_block14_0_bn[0][0]
conv5_block14_1_conv (Conv2D)	(None, 8, 8, 128)	135168	conv5_block14_0_relu[0][0]
conv5_block14_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block14_1_conv[0][0]
conv5_block14_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block14_1_bn[0][0]
conv5_block14_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block14_1_relu[0][0]
conv5_block14_concat (Concatena	(None, 8, 8, 1088)	0	conv5_block13_concat[0][0] conv5_block14_2_conv[0][0]
conv5_block15_0_bn (BatchNormal	(None, 8, 8, 1088)	4352	conv5_block14_concat[0][0]
conv5_block15_0_relu (Activatio	(None, 8, 8, 1088)	0	conv5_block15_0_bn[0][0]
conv5_block15_1_conv (Conv2D)	(None, 8, 8, 128)	139264	conv5_block15_0_relu[0][0]
conv5_block15_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block15_1_conv[0][0]
conv5_block15_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block15_1_bn[0][0]
conv5_block15_concat (Concatena	(None, 8, 8, 1120)	0	conv5_block14_concat[0][0] conv5_block15_2_conv[0][0]
conv5_block16_0_bn (BatchNormal	(None, 8, 8, 1120)	4480	conv5_block15_concat[0][0]
conv5_block16_0_relu (Activatio	(None, 8, 8, 1120)	0	conv5_block16_0_bn[0][0]
conv5_block16_1_conv (Conv2D)	(None, 8, 8, 128)	143360	conv5_block16_0_relu[0][0]
conv5_block16_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block16_1_conv[0][0]
conv5_block16_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block16_1_bn[0][0]
conv5_block16_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block16_1_relu[0][0]
conv5_block16_concat (Concatena	(None, 8, 8, 1152)	0	conv5_block15_concat[0][0] conv5_block16_2_conv[0][0]
conv5_block17_0_bn (BatchNormal	(None, 8, 8, 1152)	4608	conv5_block16_concat[0][0]
conv5_block17_0_relu (Activatio	(None, 8, 8, 1152)	0	conv5_block17_0_bn[0][0]
conv5_block17_1_conv (Conv2D)	(None, 8, 8, 128)	147456	conv5_block17_0_relu[0][0]
conv5_block17_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block17_1_conv[0][0]
conv5_block17_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block17_1_bn[0][0]
conv5_block17_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block17_1_relu[0][0]
conv5_block17_concat (Concatena	(None, 8, 8, 1184)	0	conv5_block16_concat[0][0] conv5_block17_2_conv[0][0]
conv5_block18_0_bn (BatchNormal	(None, 8, 8, 1184)	4736	conv5_block17_concat[0][0]

conv5_block15_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block15_1_relu[0][0]
conv5_block18_0_relu (Activatio	(None, 8, 8, 1184)	0	conv5_block18_0_bn[0][0]
conv5_block18_1_conv (Conv2D)	(None, 8, 8, 128)	151552	conv5_block18_0_relu[0][0]
conv5_block18_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block18_1_conv[0][0]
conv5_block18_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block18_1_bn[0][0]
conv5_block18_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block18_1_relu[0][0]
conv5_block18_concat (Concatena	(None, 8, 8, 1216)	0	conv5_block17_concat[0][0] conv5_block18_2_conv[0][0]
conv5_block19_0_bn (BatchNormal	(None, 8, 8, 1216)	4864	conv5_block18_concat[0][0]
conv5_block19_0_relu (Activatio	(None, 8, 8, 1216)	0	conv5_block19_0_bn[0][0]
conv5_block19_1_conv (Conv2D)	(None, 8, 8, 128)	155648	conv5_block19_0_relu[0][0]
conv5_block19_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block19_1_conv[0][0]
conv5_block19_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block19_1_bn[0][0]
conv5_block19_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block19_1_relu[0][0]
conv5_block19_concat (Concatena	(None, 8, 8, 1248)	0	conv5_block18_concat[0][0] conv5_block19_2_conv[0][0]
conv5_block20_0_bn (BatchNormal	(None, 8, 8, 1248)	4992	conv5_block19_concat[0][0]
conv5_block20_0_relu (Activatio	(None, 8, 8, 1248)	0	conv5_block20_0_bn[0][0]
conv5_block20_1_conv (Conv2D)	(None, 8, 8, 128)	159744	conv5_block20_0_relu[0][0]
conv5_block20_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block20_1_bn[0][0]
conv5_block20_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block20_1_relu[0][0]
conv5_block20_concat (Concatena	(None, 8, 8, 1280)	0	conv5_block19_concat[0][0] conv5_block20_2_conv[0][0]
conv5_block21_0_bn (BatchNormal	(None, 8, 8, 1280)	5120	conv5_block20_concat[0][0]
conv5_block21_0_relu (Activatio	(None, 8, 8, 1280)	0	conv5_block21_0_bn[0][0]
conv5_block21_1_conv (Conv2D)	(None, 8, 8, 128)	163840	conv5_block21_0_relu[0][0]
conv5_block21_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block21_1_conv[0][0]
conv5_block21_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block21_1_bn[0][0]
conv5_block21_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block21_1_relu[0][0]
conv5_block21_concat (Concatena	(None, 8, 8, 1312)	0	conv5_block20_concat[0][0] conv5_block21_2_conv[0][0]
conv5_block22_0_bn (BatchNormal	(None, 8, 8, 1312)	5248	conv5_block21_concat[0][0]
conv5_block22_0_relu (Activatio	(None, 8, 8, 1312)	0	conv5_block22_0_bn[0][0]
conv5_block22_1_conv (Conv2D)	(None, 8, 8, 128)	167936	conv5_block22_0_relu[0][0]
conv5_block22_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block22_1_conv[0][0]
conv5_block22_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block22_1_bn[0][0]
conv5_block22_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block22_1_relu[0][0]
conv5_block22_concat (Concatena	(None, 8, 8, 1344)	0	conv5_block21_concat[0][0] conv5_block22_2_conv[0][0]
conv5_block23_0_bn (BatchNormal	(None, 8, 8, 1344)	5376	conv5_block22_concat[0][0]
conv5_block23_0_relu (Activatio	(None, 8, 8, 1344)	0	conv5_block23_0_bn[0][0]
conv5_block23_1_conv (Conv2D)	(None, 8, 8, 128)	172032	conv5_block23_0_relu[0][0]
conv5_block23_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block23_1_conv[0][0]
conv5_block23_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block23_1_bn[0][0]
conv5_block23_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block23_1_relu[0][0]
conv5_block23_concat (Concatena	(None, 8, 8, 1376)	0	conv5_block22_concat[0][0] conv5_block23_2_conv[0][0]
conv5_block24_0_bn (BatchNormal	(None, 8, 8, 1376)	5504	conv5_block23_concat[0][0]
conv5_block24_0_relu (Activatio	(None, 8, 8, 1376)	0	conv5_block24_0_bn[0][0]
conv5_block24_1_conv (Conv2D)	(None, 8, 8, 128)	176128	conv5_block24_0_relu[0][0]
conv5_block24_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block24_1_conv[0][0]
conv5_block24_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block24_1_bn[0][0]
conv5_block24_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block24_1_relu[0][0]
conv5_block24_concat (Concatena	(None, 8, 8, 1408)	0	conv5_block23_concat[0][0] conv5_block24_2_conv[0][0]
conv5_block20_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block20_1_conv[0][0]
conv5_block25_1_conv (Conv2D)	(None, 8, 8, 128)	180224	conv5_block25_0_relu[0][0]
conv5_block25_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block25_1_conv[0][0]
conv5_block25_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block25_1_bn[0][0]
conv5_block25_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block25_1_relu[0][0]
conv5_block25_concat (Concatena	(None, 8, 8, 1440)	0	conv5_block24_concat[0][0] conv5_block25_2_conv[0][0]
conv5_block26_0_bn (BatchNormal	(None, 8, 8, 1440)	5760	conv5_block25_concat[0][0]
conv5_block26_0_relu (Activatio	(None, 8, 8, 1440)	0	conv5_block26_0_bn[0][0]
conv5_block26_1_conv (Conv2D)	(None, 8, 8, 128)	184320	conv5_block26_0_relu[0][0]
conv5_block26_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block26_1_conv[0][0]
conv5_block26_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block26_1_bn[0][0]
conv5_block26_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block26_1_relu[0][0]
conv5_block26_concat (Concatena	(None, 8, 8, 1472)	0	conv5_block25_concat[0][0] conv5_block26_2_conv[0][0]
conv5_block27_0_bn (BatchNormal	(None, 8, 8, 1472)	5888	conv5_block26_concat[0][0]
conv5_block27_0_relu (Activatio	(None, 8, 8, 1472)	0	conv5_block27_0_bn[0][0]
conv5_block27_1_conv (Conv2D)	(None, 8, 8, 128)	188416	conv5_block27_0_relu[0][0]
conv5_block27_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block27_1_conv[0][0]
conv5_block27_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block27_1_bn[0][0]
conv5_block27_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block27_1_relu[0][0]
conv5_block27_concat (Concatena	(None, 8, 8, 1504)	0	conv5_block26_concat[0][0] conv5_block27_2_conv[0][0]
conv5_block28_0_bn (BatchNormal	(None, 8, 8, 1504)	6016	conv5_block27_concat[0][0]
conv5_block28_0_relu (Activatio	(None, 8, 8, 1504)	0	conv5_block28_0_bn[0][0]
conv5_block28_1_conv (Conv2D)	(None, 8, 8, 128)	192512	conv5_block28_0_relu[0][0]
conv5_block28_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block28_1_conv[0][0]
conv5_block28_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block28_1_bn[0][0]
conv5_block28_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block28_1_relu[0][0]
conv5_block28_concat (Concatena	(None, 8, 8, 1536)	0	conv5_block27_concat[0][0] conv5_block28_2_conv[0][0]
conv5_block29_0_bn (BatchNormal	(None, 8, 8, 1536)	6144	conv5_block28_concat[0][0]
conv5_block29_0_relu (Activatio	(None, 8, 8, 1536)	0	conv5_block29_0_bn[0][0]
conv5_block29_1_conv (Conv2D)	(None, 8, 8, 128)	196608	conv5_block29_0_relu[0][0]
conv5_block29_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block29_1_conv[0][0]
conv5_block29_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block29_1_bn[0][0]
conv5_block29_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block29_1_relu[0][0]
conv5_block25_0_bn (BatchNormal	(None, 8, 8, 1408)	5632	conv5_block24_concat[0][0]
conv5_block25_0_relu (Activatio	(None, 8, 8, 1408)	0	conv5_block25_0_bn[0][0]
conv5_block29_concat (Concatena	(None, 8, 8, 1568)	0	conv5_block28_concat[0][0] conv5_block29_2_conv[0][0]
conv5_block30_0_bn (BatchNormal	(None, 8, 8, 1568)	6272	conv5_block29_concat[0][0]
conv5_block30_0_relu (Activatio	(None, 8, 8, 1568)	0	conv5_block30_0_bn[0][0]
conv5_block30_1_conv (Conv2D)	(None, 8, 8, 128)	200704	conv5_block30_0_relu[0][0]
conv5_block30_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block30_1_conv[0][0]
conv5_block30_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block30_1_bn[0][0]
conv5_block30_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block30_1_relu[0][0]
conv5_block30_concat (Concatena	(None, 8, 8, 1600)	0	conv5_block29_concat[0][0] conv5_block30_2_conv[0][0]
conv5_block31_0_bn (BatchNormal	(None, 8, 8, 1600)	6400	conv5_block30_concat[0][0]
conv5_block31_0_relu (Activatio	(None, 8, 8, 1600)	0	conv5_block31_0_bn[0][0]
conv5_block31_1_conv (Conv2D)	(None, 8, 8, 128)	204800	conv5_block31_0_relu[0][0]
conv5_block31_1_bn (BatchNormal	(None, 8, 8, 128)	512	conv5_block31_1_conv[0][0]
conv5_block31_1_relu (Activatio	(None, 8, 8, 128)	0	conv5_block31_1_bn[0][0]
conv5_block31_2_conv (Conv2D)	(None, 8, 8, 32)	36864	conv5_block31_1_relu[0][0]
conv5_block31_concat (Concatena	(None, 8, 8, 1632)	0	conv5_block30_concat[0][0] conv5_block31_2_conv[0][0]
conv5_block32_0_bn (BatchNormal	(None, 8, 8, 1632)	6528	conv5_block31_concat[0][0]
conv5_block32_0_relu (Activatio	(None, 8, 8, 1632)	0	conv5_block32_0_bn[0][0]

ResNet152V2

In [14]:

```
base_model = ResNet152V2(input_shape=(256,256,3),include_top=False)
for layer in base_model.layers:
    layer.trainable=False

X=Flatten()(base_model.output)
X=Dense(units=2, activation='sigmoid')(X)

model_RN = Model(base_model.input, X)

model_RN.compile(optimizer='adam',loss=keras.losses.binary_crossentropy,metrics=['accuracy'])
model_RN.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_2 (InputLayer)            [(None, 256, 256, 3) 0
__________________________________________________________________________________________________
conv1_pad (ZeroPadding2D)       (None, 262, 262, 3)  0           input_2[0][0]
__________________________________________________________________________________________________
conv1_conv (Conv2D)             (None, 128, 128, 64) 9472        conv1_pad[0][0]
__________________________________________________________________________________________________
pool1_pad (ZeroPadding2D)       (None, 130, 130, 64) 0           conv1_conv[0][0]
__________________________________________________________________________________________________
pool1_pool (MaxPooling2D)       (None, 64, 64, 64)   0           pool1_pad[0][0]
__________________________________________________________________________________________________
conv2_block1_preact_bn (BatchNo (None, 64, 64, 64)   256         pool1_pool[0][0]
__________________________________________________________________________________________________
conv2_block1_preact_relu (Activ (None, 64, 64, 64)   0           conv2_block1_preact_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_1_conv (Conv2D)    (None, 64, 64, 64)   4096        conv2_block1_preact_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_1_bn (BatchNormali (None, 64, 64, 64)   256         conv2_block1_1_conv[0][0]
__________________________________________________________________________________________________
conv2_block1_1_relu (Activation (None, 64, 64, 64)   0           conv2_block1_1_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_2_pad (ZeroPadding (None, 66, 66, 64)   0           conv2_block1_1_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_2_conv (Conv2D)    (None, 64, 64, 64)   36864       conv2_block1_2_pad[0][0]
__________________________________________________________________________________________________
conv2_block1_2_bn (BatchNormali (None, 64, 64, 64)   256         conv2_block1_2_conv[0][0]
__________________________________________________________________________________________________
conv2_block1_2_relu (Activation (None, 64, 64, 64)   0           conv2_block1_2_bn[0][0]
__________________________________________________________________________________________________
conv2_block1_0_conv (Conv2D)    (None, 64, 64, 256)  16640       conv2_block1_preact_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_3_conv (Conv2D)    (None, 64, 64, 256)  16640       conv2_block1_2_relu[0][0]
__________________________________________________________________________________________________
conv2_block1_out (Add)          (None, 64, 64, 256)  0           conv2_block1_0_conv[0][0]
                                                                 conv2_block1_3_conv[0][0]
__________________________________________________________________________________________________
conv2_block2_preact_bn (BatchNo (None, 64, 64, 256)  1024        conv2_block1_out[0][0]
__________________________________________________________________________________________________
conv2_block2_preact_relu (Activ (None, 64, 64, 256)  0           conv2_block2_preact_bn[0][0]
__________________________________________________________________________________________________
conv2_block2_1_conv (Conv2D)    (None, 64, 64, 64)   16384       conv2_block2_preact_relu[0][0]
__________________________________________________________________________________________________
conv2_block2_1_bn (BatchNormali (None, 64, 64, 64)   256         conv2_block2_1_conv[0][0]
__________________________________________________________________________________________________
conv2_block2_1_relu (Activation (None, 64, 64, 64)   0           conv2_block2_1_bn[0][0]
__________________________________________________________________________________________________
conv2_block2_2_pad (ZeroPadding (None, 66, 66, 64)   0           conv2_block2_1_relu[0][0]
__________________________________________________________________________________________________
conv2_block2_2_conv (Conv2D)    (None, 64, 64, 64)   36864       conv2_block2_2_pad[0][0]
```

conv2_block2_2_bn (BatchNormali	(None, 64, 64, 64)	256	conv2_block2_2_conv[0][0]
conv2_block2_2_relu (Activation	(None, 64, 64, 64)	0	conv2_block2_2_bn[0][0]
conv2_block2_3_conv (Conv2D)	(None, 64, 64, 256)	16640	conv2_block2_2_relu[0][0]
conv2_block2_out (Add)	(None, 64, 64, 256)	0	conv2_block1_out[0][0] conv2_block2_3_conv[0][0]
conv2_block3_preact_bn (BatchNo	(None, 64, 64, 256)	1024	conv2_block2_out[0][0]
conv2_block3_preact_relu (Activ	(None, 64, 64, 256)	0	conv2_block3_preact_bn[0][0]
conv2_block3_1_conv (Conv2D)	(None, 64, 64, 64)	16384	conv2_block3_preact_relu[0][0]
conv2_block3_1_bn (BatchNormali	(None, 64, 64, 64)	256	conv2_block3_1_conv[0][0]
conv2_block3_1_relu (Activation	(None, 64, 64, 64)	0	conv2_block3_1_bn[0][0]
conv2_block3_2_pad (ZeroPadding	(None, 66, 66, 64)	0	conv2_block3_1_relu[0][0]
conv2_block3_2_conv (Conv2D)	(None, 32, 32, 64)	36864	conv2_block3_2_pad[0][0]
conv2_block3_2_bn (BatchNormali	(None, 32, 32, 64)	256	conv2_block3_2_conv[0][0]
conv2_block3_2_relu (Activation	(None, 32, 32, 64)	0	conv2_block3_2_bn[0][0]
max_pooling2d (MaxPooling2D)	(None, 32, 32, 256)	0	conv2_block2_out[0][0]
conv2_block3_3_conv (Conv2D)	(None, 32, 32, 256)	16640	conv2_block3_2_relu[0][0]
conv2_block3_out (Add)	(None, 32, 32, 256)	0	max_pooling2d[0][0] conv2_block3_3_conv[0][0]
conv3_block1_preact_bn (BatchNo	(None, 32, 32, 256)	1024	conv2_block3_out[0][0]
conv3_block1_preact_relu (Activ	(None, 32, 32, 256)	0	conv3_block1_preact_bn[0][0]
conv3_block1_1_conv (Conv2D)	(None, 32, 32, 128)	32768	conv3_block1_preact_relu[0][0]
conv3_block1_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block1_1_conv[0][0]
conv3_block1_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block1_1_bn[0][0]
conv3_block1_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block1_1_relu[0][0]
conv3_block1_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block1_2_pad[0][0]
conv3_block1_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block1_2_conv[0][0]
conv3_block1_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block1_2_bn[0][0]
conv3_block1_0_conv (Conv2D)	(None, 32, 32, 512)	131584	conv3_block1_preact_relu[0][0]
conv3_block1_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block1_2_relu[0][0]
conv3_block1_out (Add)	(None, 32, 32, 512)	0	conv3_block1_0_conv[0][0] conv3_block1_3_conv[0][0]
conv3_block2_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block1_out[0][0]
conv3_block2_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block2_preact_bn[0][0]
conv3_block2_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block2_preact_relu[0][0]
conv3_block2_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block2_1_conv[0][0]
conv3_block2_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block2_1_bn[0][0]
conv3_block2_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block2_1_relu[0][0]
conv3_block2_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block2_2_pad[0][0]
conv3_block2_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block2_2_conv[0][0]
conv3_block2_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block2_2_bn[0][0]
conv3_block2_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block2_2_relu[0][0]
conv3_block2_out (Add)	(None, 32, 32, 512)	0	conv3_block1_out[0][0] conv3_block2_3_conv[0][0]
conv3_block3_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block2_out[0][0]
conv3_block3_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block3_preact_bn[0][0]
conv3_block3_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block3_preact_relu[0][0]
conv3_block3_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block3_1_conv[0][0]
conv3_block3_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block3_1_bn[0][0]
conv3_block3_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block3_1_relu[0][0]
conv3_block3_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block3_2_pad[0][0]
conv3_block3_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block3_2_conv[0][0]
conv3_block3_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block3_2_bn[0][0]
conv3_block3_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block3_2_relu[0][0]
conv3_block3_out (Add)	(None, 32, 32, 512)	0	conv3_block2_out[0][0] conv3_block3_3_conv[0][0]
conv3_block4_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block3_out[0][0]
conv3_block4_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block4_preact_bn[0][0]
conv3_block4_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block4_preact_relu[0][0]
conv3_block4_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block4_1_conv[0][0]
conv3_block4_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block4_1_bn[0][0]
conv3_block4_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block4_1_relu[0][0]
conv3_block4_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block4_2_pad[0][0]
conv3_block4_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block4_2_conv[0][0]
conv3_block4_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block4_2_bn[0][0]
conv3_block4_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block4_2_relu[0][0]
conv3_block4_out (Add)	(None, 32, 32, 512)	0	conv3_block3_out[0][0] conv3_block4_3_conv[0][0]
conv3_block5_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block4_out[0][0]
conv3_block5_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block5_preact_bn[0][0]
conv3_block5_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block5_preact_relu[0][0]
conv3_block5_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block5_1_conv[0][0]
conv3_block5_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block5_1_bn[0][0]
conv3_block5_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block5_1_relu[0][0]
conv3_block5_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block5_2_pad[0][0]
conv3_block5_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block5_2_conv[0][0]
conv3_block5_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block5_2_bn[0][0]
conv3_block5_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block5_2_relu[0][0]
conv3_block5_out (Add)	(None, 32, 32, 512)	0	conv3_block4_out[0][0] conv3_block5_3_conv[0][0]
conv3_block6_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block5_out[0][0]
conv3_block6_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block6_preact_bn[0][0]
conv3_block6_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block6_preact_relu[0][0]
conv3_block6_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block6_1_conv[0][0]
conv3_block6_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block6_1_bn[0][0]
conv3_block6_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block6_1_relu[0][0]
conv3_block6_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block6_2_pad[0][0]
conv3_block6_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block6_2_conv[0][0]
conv3_block6_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block6_2_bn[0][0]
conv3_block6_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block6_2_relu[0][0]
conv3_block6_out (Add)	(None, 32, 32, 512)	0	conv3_block5_out[0][0] conv3_block6_3_conv[0][0]
conv3_block7_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block6_out[0][0]
conv3_block7_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block7_preact_bn[0][0]
conv3_block7_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block7_preact_relu[0][0]
conv3_block7_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block7_1_conv[0][0]
conv3_block7_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block7_1_bn[0][0]
conv3_block7_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block7_1_relu[0][0]

conv3_block7_2_conv (Conv2D)	(None, 32, 32, 128)	147456	conv3_block7_2_pad[0][0]
conv3_block7_2_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block7_2_conv[0][0]
conv3_block7_2_relu (Activation	(None, 32, 32, 128)	0	conv3_block7_2_bn[0][0]
conv3_block7_3_conv (Conv2D)	(None, 32, 32, 512)	66048	conv3_block7_2_relu[0][0]
conv3_block7_out (Add)	(None, 32, 32, 512)	0	conv3_block6_out[0][0] conv3_block7_3_conv[0][0]
conv3_block8_preact_bn (BatchNo	(None, 32, 32, 512)	2048	conv3_block7_out[0][0]
conv3_block8_preact_relu (Activ	(None, 32, 32, 512)	0	conv3_block8_preact_bn[0][0]
conv3_block8_1_conv (Conv2D)	(None, 32, 32, 128)	65536	conv3_block8_preact_relu[0][0]
conv3_block8_1_bn (BatchNormali	(None, 32, 32, 128)	512	conv3_block8_1_conv[0][0]
conv3_block8_1_relu (Activation	(None, 32, 32, 128)	0	conv3_block8_1_bn[0][0]
conv3_block8_2_pad (ZeroPadding	(None, 34, 34, 128)	0	conv3_block8_1_relu[0][0]
conv3_block8_2_conv (Conv2D)	(None, 16, 16, 128)	147456	conv3_block8_2_pad[0][0]
conv3_block8_2_bn (BatchNormali	(None, 16, 16, 128)	512	conv3_block8_2_conv[0][0]
conv3_block8_2_relu (Activation	(None, 16, 16, 128)	0	conv3_block8_2_bn[0][0]
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 512)	0	conv3_block7_out[0][0]
conv3_block8_3_conv (Conv2D)	(None, 16, 16, 512)	66048	conv3_block8_2_relu[0][0]
conv3_block8_out (Add)	(None, 16, 16, 512)	0	max_pooling2d_1[0][0] conv3_block8_3_conv[0][0]
conv4_block1_preact_bn (BatchNo	(None, 16, 16, 512)	2048	conv3_block8_out[0][0]
conv4_block1_preact_relu (Activ	(None, 16, 16, 512)	0	conv4_block1_preact_bn[0][0]
conv4_block1_1_conv (Conv2D)	(None, 16, 16, 256)	131072	conv4_block1_preact_relu[0][0]
conv4_block1_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block1_1_conv[0][0]
conv4_block1_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block1_1_bn[0][0]
conv4_block1_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block1_1_relu[0][0]
conv4_block1_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block1_2_pad[0][0]
conv4_block1_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block1_2_conv[0][0]
conv4_block1_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block1_2_bn[0][0]
conv4_block1_0_conv (Conv2D)	(None, 16, 16, 1024)	525312	conv4_block1_preact_relu[0][0]
conv4_block1_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block1_2_relu[0][0]
conv4_block1_out (Add)	(None, 16, 16, 1024)	0	conv4_block1_0_conv[0][0] conv4_block1_3_conv[0][0]
conv4_block2_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block1_out[0][0]
conv4_block2_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block2_preact_bn[0][0]
conv4_block2_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block2_preact_relu[0][0]
conv4_block2_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block2_1_conv[0][0]
conv4_block2_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block2_1_bn[0][0]
conv4_block2_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block2_1_relu[0][0]
conv4_block2_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block2_2_pad[0][0]
conv4_block2_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block2_2_conv[0][0]
conv4_block2_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block2_2_bn[0][0]
conv4_block2_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block2_2_relu[0][0]
conv4_block2_out (Add)	(None, 16, 16, 1024)	0	conv4_block1_out[0][0] conv4_block2_3_conv[0][0]
conv4_block3_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block2_out[0][0]
conv4_block3_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block3_preact_bn[0][0]
conv4_block3_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block3_preact_relu[0][0]
conv4_block3_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block3_1_conv[0][0]
conv4_block3_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block3_1_bn[0][0]
conv4_block3_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block3_1_relu[0][0]
conv4_block3_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block3_2_pad[0][0]
conv4_block3_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block3_2_conv[0][0]
conv4_block3_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block3_2_bn[0][0]
conv4_block3_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block3_2_relu[0][0]
conv4_block3_out (Add)	(None, 16, 16, 1024)	0	conv4_block2_out[0][0] conv4_block3_3_conv[0][0]
conv4_block4_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block3_out[0][0]
conv4_block4_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block4_preact_bn[0][0]
conv4_block4_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block4_preact_relu[0][0]
conv4_block4_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block4_1_conv[0][0]
conv4_block4_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block4_1_bn[0][0]
conv4_block4_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block4_1_relu[0][0]
conv4_block4_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block4_2_pad[0][0]
conv4_block4_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block4_2_conv[0][0]
conv4_block4_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block4_2_bn[0][0]
conv4_block4_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block4_2_relu[0][0]
conv4_block4_out (Add)	(None, 16, 16, 1024)	0	conv4_block3_out[0][0] conv4_block4_3_conv[0][0]
conv4_block5_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block4_out[0][0]
conv4_block5_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block5_preact_bn[0][0]
conv4_block5_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block5_preact_relu[0][0]
conv4_block5_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block5_1_conv[0][0]
conv4_block5_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block5_1_bn[0][0]
conv4_block5_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block5_1_relu[0][0]
conv4_block5_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block5_2_pad[0][0]
conv4_block5_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block5_2_conv[0][0]
conv4_block5_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block5_2_bn[0][0]
conv4_block5_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block5_2_relu[0][0]
conv4_block5_out (Add)	(None, 16, 16, 1024)	0	conv4_block4_out[0][0] conv4_block5_3_conv[0][0]
conv4_block6_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block5_out[0][0]
conv4_block6_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block6_preact_bn[0][0]
conv4_block6_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block6_preact_relu[0][0]
conv4_block6_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block6_1_conv[0][0]
conv4_block6_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block6_1_bn[0][0]
conv4_block6_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block6_1_relu[0][0]
conv4_block6_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block6_2_pad[0][0]
conv4_block6_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block6_2_conv[0][0]
conv4_block6_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block6_2_bn[0][0]
conv4_block6_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block6_2_relu[0][0]
conv4_block6_out (Add)	(None, 16, 16, 1024)	0	conv4_block5_out[0][0] conv4_block6_3_conv[0][0]
conv4_block7_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block6_out[0][0]
conv4_block7_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block7_preact_bn[0][0]
conv4_block7_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block7_preact_relu[0][0]
conv4_block7_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block7_1_conv[0][0]
conv4_block7_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block7_1_bn[0][0]
conv4_block7_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block7_1_relu[0][0]
conv4_block7_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block7_2_pad[0][0]
conv4_block7_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block7_2_conv[0][0]
conv4_block7_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block7_2_bn[0][0]
conv4_block7_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block7_2_relu[0][0]
conv4_block7_out (Add)	(None, 16, 16, 1024)	0	conv4_block6_out[0][0] conv4_block7_3_conv[0][0]

conv4_block8_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block7_out[0][0]
conv4_block8_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block8_preact_bn[0][0]
conv4_block8_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block8_preact_relu[0][0]
conv4_block8_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block8_1_conv[0][0]
conv4_block8_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block8_1_bn[0][0]
conv4_block8_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block8_1_relu[0][0]
conv4_block8_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block8_2_pad[0][0]
conv4_block8_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block8_2_conv[0][0]
conv4_block8_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block8_2_bn[0][0]
conv4_block8_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block8_2_relu[0][0]
conv4_block8_out (Add)	(None, 16, 16, 1024)	0	conv4_block7_out[0][0] conv4_block8_3_conv[0][0]
conv4_block9_preact_bn (BatchNo	(None, 16, 16, 1024)	4096	conv4_block8_out[0][0]
conv4_block9_preact_relu (Activ	(None, 16, 16, 1024)	0	conv4_block9_preact_bn[0][0]
conv4_block9_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block9_preact_relu[0][0]
conv4_block9_1_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block9_1_conv[0][0]
conv4_block9_1_relu (Activation	(None, 16, 16, 256)	0	conv4_block9_1_bn[0][0]
conv4_block9_2_pad (ZeroPadding	(None, 18, 18, 256)	0	conv4_block9_1_relu[0][0]
conv4_block9_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block9_2_pad[0][0]
conv4_block9_2_bn (BatchNormali	(None, 16, 16, 256)	1024	conv4_block9_2_conv[0][0]
conv4_block9_2_relu (Activation	(None, 16, 16, 256)	0	conv4_block9_2_bn[0][0]
conv4_block9_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block9_2_relu[0][0]
conv4_block9_out (Add)	(None, 16, 16, 1024)	0	conv4_block8_out[0][0] conv4_block9_3_conv[0][0]
conv4_block10_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block9_out[0][0]
conv4_block10_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block10_preact_bn[0][0]
conv4_block10_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block10_preact_relu[0][0]
conv4_block10_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block10_1_conv[0][0]
conv4_block10_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block10_1_bn[0][0]
conv4_block10_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block10_1_relu[0][0]
conv4_block10_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block10_2_pad[0][0]
conv4_block10_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block10_2_conv[0][0]
conv4_block10_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block10_2_bn[0][0]
conv4_block10_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block10_2_relu[0][0]
conv4_block10_out (Add)	(None, 16, 16, 1024)	0	conv4_block9_out[0][0] conv4_block10_3_conv[0][0]
conv4_block11_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block10_out[0][0]
conv4_block11_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block11_preact_bn[0][0]
conv4_block11_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block11_preact_relu[0][0]
conv4_block11_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block11_1_conv[0][0]
conv4_block11_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block11_1_bn[0][0]
conv4_block11_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block11_1_relu[0][0]
conv4_block11_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block11_2_pad[0][0]
conv4_block11_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block11_2_conv[0][0]
conv4_block11_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block11_2_bn[0][0]
conv4_block11_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block11_2_relu[0][0]
conv4_block11_out (Add)	(None, 16, 16, 1024)	0	conv4_block10_out[0][0] conv4_block11_3_conv[0][0]
conv4_block12_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block11_out[0][0]
conv4_block12_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block12_preact_bn[0][0]
conv4_block12_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block12_preact_relu[0][0]
conv4_block12_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block12_1_conv[0][0]
conv4_block12_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block12_1_bn[0][0]
conv4_block12_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block12_1_relu[0][0]
conv4_block12_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block12_2_pad[0][0]
conv4_block12_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block12_2_conv[0][0]
conv4_block12_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block12_2_bn[0][0]
conv4_block12_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block12_2_relu[0][0]
conv4_block12_out (Add)	(None, 16, 16, 1024)	0	conv4_block11_out[0][0] conv4_block12_3_conv[0][0]
conv4_block13_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block12_out[0][0]
conv4_block13_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block13_preact_bn[0][0]
conv4_block13_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block13_preact_relu[0][0]
conv4_block13_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block13_1_conv[0][0]
conv4_block13_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block13_1_bn[0][0]
conv4_block13_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block13_1_relu[0][0]
conv4_block13_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block13_2_pad[0][0]
conv4_block13_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block13_2_conv[0][0]
conv4_block13_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block13_2_bn[0][0]
conv4_block13_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block13_2_relu[0][0]
conv4_block13_out (Add)	(None, 16, 16, 1024)	0	conv4_block12_out[0][0] conv4_block13_3_conv[0][0]
conv4_block14_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block13_out[0][0]
conv4_block14_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block14_preact_bn[0][0]
conv4_block14_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block14_preact_relu[0][0]
conv4_block14_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block14_1_conv[0][0]
conv4_block14_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block14_1_bn[0][0]
conv4_block14_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block14_1_relu[0][0]
conv4_block14_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block14_2_pad[0][0]
conv4_block14_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block14_2_conv[0][0]
conv4_block14_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block14_2_bn[0][0]
conv4_block14_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block14_2_relu[0][0]
conv4_block14_out (Add)	(None, 16, 16, 1024)	0	conv4_block13_out[0][0] conv4_block14_3_conv[0][0]
conv4_block15_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block14_out[0][0]
conv4_block15_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block15_preact_bn[0][0]
conv4_block15_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block15_preact_relu[0][0]
conv4_block15_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block15_1_conv[0][0]
conv4_block15_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block15_1_bn[0][0]
conv4_block15_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block15_1_relu[0][0]
conv4_block15_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block15_2_pad[0][0]
conv4_block15_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block15_2_conv[0][0]
conv4_block15_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block15_2_bn[0][0]
conv4_block15_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block15_2_relu[0][0]
conv4_block15_out (Add)	(None, 16, 16, 1024)	0	conv4_block14_out[0][0] conv4_block15_3_conv[0][0]
conv4_block16_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block15_out[0][0]
conv4_block16_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block16_preact_bn[0][0]
conv4_block16_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block16_preact_relu[0][0]
conv4_block16_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block16_1_conv[0][0]
conv4_block16_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block16_1_bn[0][0]
conv4_block16_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block16_1_relu[0][0]
conv4_block16_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block16_2_pad[0][0]
conv4_block16_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block16_2_conv[0][0]
conv4_block16_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block16_2_bn[0][0]
conv4_block16_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block16_2_relu[0][0]
conv4_block16_out (Add)	(None, 16, 16, 1024)	0	conv4_block15_out[0][0] conv4_block16_3_conv[0][0]
conv4_block17_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block16_out[0][0]
conv4_block17_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block17_preact_bn[0][0]
conv4_block17_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block17_preact_relu[0][0]
conv4_block17_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block17_1_conv[0][0]
conv4_block17_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block17_1_bn[0][0]
conv4_block17_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block17_1_relu[0][0]
conv4_block17_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block17_2_pad[0][0]
conv4_block17_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block17_2_conv[0][0]
conv4_block17_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block17_2_bn[0][0]
conv4_block17_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block17_2_relu[0][0]
conv4_block17_out (Add)	(None, 16, 16, 1024)	0	conv4_block16_out[0][0] conv4_block17_3_conv[0][0]

conv4_block18_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block17_out[0][0]
conv4_block18_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block18_preact_bn[0][0]
conv4_block18_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block18_preact_relu[0][0]
conv4_block18_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block18_1_conv[0][0]
conv4_block18_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block18_1_bn[0][0]
conv4_block18_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block18_1_relu[0][0]
conv4_block18_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block18_2_pad[0][0]
conv4_block18_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block18_2_conv[0][0]
conv4_block18_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block18_2_bn[0][0]
conv4_block18_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block18_2_relu[0][0]
conv4_block18_out (Add)	(None, 16, 16, 1024)	0	conv4_block17_out[0][0] conv4_block18_3_conv[0][0]
conv4_block19_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block18_out[0][0]
conv4_block19_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block19_preact_bn[0][0]
conv4_block19_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block19_preact_relu[0][0]
conv4_block19_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block19_1_conv[0][0]
conv4_block19_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block19_1_bn[0][0]
conv4_block19_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block19_1_relu[0][0]
conv4_block19_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block19_2_pad[0][0]
conv4_block19_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block19_2_conv[0][0]
conv4_block19_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block19_2_bn[0][0]
conv4_block19_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block19_2_relu[0][0]
conv4_block19_out (Add)	(None, 16, 16, 1024)	0	conv4_block18_out[0][0] conv4_block19_3_conv[0][0]
conv4_block20_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block19_out[0][0]
conv4_block20_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block20_preact_bn[0][0]
conv4_block20_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block20_preact_relu[0][0]
conv4_block20_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block20_1_conv[0][0]
conv4_block20_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block20_1_bn[0][0]
conv4_block20_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block20_1_relu[0][0]
conv4_block20_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block20_2_pad[0][0]
conv4_block20_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block20_2_conv[0][0]
conv4_block20_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block20_2_bn[0][0]
conv4_block20_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block20_2_relu[0][0]
conv4_block20_out (Add)	(None, 16, 16, 1024)	0	conv4_block19_out[0][0] conv4_block20_3_conv[0][0]
conv4_block21_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block20_out[0][0]
conv4_block21_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block21_preact_bn[0][0]
conv4_block21_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block21_preact_relu[0][0]
conv4_block21_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block21_1_conv[0][0]
conv4_block21_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block21_1_bn[0][0]
conv4_block21_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block21_1_relu[0][0]
conv4_block21_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block21_2_pad[0][0]
conv4_block21_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block21_2_conv[0][0]
conv4_block21_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block21_2_bn[0][0]
conv4_block21_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block21_2_relu[0][0]
conv4_block21_out (Add)	(None, 16, 16, 1024)	0	conv4_block20_out[0][0] conv4_block21_3_conv[0][0]
conv4_block22_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block21_out[0][0]
conv4_block22_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block22_preact_bn[0][0]
conv4_block22_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block22_preact_relu[0][0]
conv4_block22_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block22_1_conv[0][0]
conv4_block22_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block22_1_bn[0][0]
conv4_block22_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block22_1_relu[0][0]
conv4_block22_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block22_2_pad[0][0]
conv4_block22_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block22_2_conv[0][0]
conv4_block22_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block22_2_bn[0][0]
conv4_block22_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block22_2_relu[0][0]
conv4_block22_out (Add)	(None, 16, 16, 1024)	0	conv4_block21_out[0][0] conv4_block22_3_conv[0][0]
conv4_block23_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block22_out[0][0]
conv4_block23_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block23_preact_bn[0][0]
conv4_block23_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block23_preact_relu[0][0]
conv4_block23_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block23_1_conv[0][0]
conv4_block23_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block23_1_bn[0][0]
conv4_block23_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block23_1_relu[0][0]
conv4_block23_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block23_2_pad[0][0]
conv4_block23_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block23_2_conv[0][0]
conv4_block23_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block23_2_bn[0][0]
conv4_block23_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block23_2_relu[0][0]
conv4_block23_out (Add)	(None, 16, 16, 1024)	0	conv4_block22_out[0][0] conv4_block23_3_conv[0][0]
conv4_block24_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block23_out[0][0]
conv4_block24_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block24_preact_bn[0][0]
conv4_block24_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block24_preact_relu[0][0]
conv4_block24_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block24_1_conv[0][0]
conv4_block24_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block24_1_bn[0][0]
conv4_block24_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block24_1_relu[0][0]
conv4_block24_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block24_2_pad[0][0]
conv4_block24_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block24_2_conv[0][0]
conv4_block24_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block24_2_bn[0][0]
conv4_block24_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block24_2_relu[0][0]
conv4_block24_out (Add)	(None, 16, 16, 1024)	0	conv4_block23_out[0][0] conv4_block24_3_conv[0][0]
conv4_block25_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block24_out[0][0]
conv4_block25_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block25_preact_bn[0][0]
conv4_block25_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block25_preact_relu[0][0]
conv4_block25_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block25_1_conv[0][0]
conv4_block25_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block25_1_bn[0][0]
conv4_block25_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block25_1_relu[0][0]
conv4_block25_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block25_2_pad[0][0]
conv4_block25_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block25_2_conv[0][0]
conv4_block25_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block25_2_bn[0][0]
conv4_block25_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block25_2_relu[0][0]
conv4_block25_out (Add)	(None, 16, 16, 1024)	0	conv4_block24_out[0][0] conv4_block25_3_conv[0][0]
conv4_block26_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block25_out[0][0]
conv4_block26_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block26_preact_bn[0][0]
conv4_block26_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block26_preact_relu[0][0]
conv4_block26_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block26_1_conv[0][0]
conv4_block26_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block26_1_bn[0][0]
conv4_block26_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block26_1_relu[0][0]
conv4_block26_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block26_2_pad[0][0]
conv4_block26_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block26_2_conv[0][0]
conv4_block26_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block26_2_bn[0][0]
conv4_block26_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block26_2_relu[0][0]
conv4_block26_out (Add)	(None, 16, 16, 1024)	0	conv4_block25_out[0][0] conv4_block26_3_conv[0][0]
conv4_block27_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block26_out[0][0]
conv4_block27_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block27_preact_bn[0][0]
conv4_block27_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block27_preact_relu[0][0]
conv4_block27_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block27_1_conv[0][0]
conv4_block27_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block27_1_bn[0][0]
conv4_block27_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block27_1_relu[0][0]
conv4_block27_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block27_2_pad[0][0]
conv4_block27_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block27_2_conv[0][0]
conv4_block27_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block27_2_bn[0][0]
conv4_block27_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block27_2_relu[0][0]

conv4_block27_out (Add)	(None, 16, 16, 1024)	0	conv4_block26_out[0][0] conv4_block27_3_conv[0][0]
conv4_block28_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block27_out[0][0]
conv4_block28_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block28_preact_bn[0][0]
conv4_block28_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block28_preact_relu[0][0]
conv4_block28_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block28_1_conv[0][0]
conv4_block28_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block28_1_bn[0][0]
conv4_block28_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block28_1_relu[0][0]
conv4_block28_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block28_2_pad[0][0]
conv4_block28_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block28_2_conv[0][0]
conv4_block28_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block28_2_bn[0][0]
conv4_block28_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block28_2_relu[0][0]
conv4_block28_out (Add)	(None, 16, 16, 1024)	0	conv4_block27_out[0][0] conv4_block28_3_conv[0][0]
conv4_block29_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block28_out[0][0]
conv4_block29_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block29_preact_bn[0][0]
conv4_block29_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block29_preact_relu[0][0]
conv4_block29_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block29_1_conv[0][0]
conv4_block29_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block29_1_bn[0][0]
conv4_block29_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block29_1_relu[0][0]
conv4_block29_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block29_2_pad[0][0]
conv4_block29_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block29_2_conv[0][0]
conv4_block29_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block29_2_bn[0][0]
conv4_block29_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block29_2_relu[0][0]
conv4_block29_out (Add)	(None, 16, 16, 1024)	0	conv4_block28_out[0][0] conv4_block29_3_conv[0][0]
conv4_block30_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block29_out[0][0]
conv4_block30_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block30_preact_bn[0][0]
conv4_block30_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block30_preact_relu[0][0]
conv4_block30_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block30_1_conv[0][0]
conv4_block30_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block30_1_bn[0][0]
conv4_block30_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block30_1_relu[0][0]
conv4_block30_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block30_2_pad[0][0]
conv4_block30_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block30_2_conv[0][0]
conv4_block30_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block30_2_bn[0][0]
conv4_block30_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block30_2_relu[0][0]
conv4_block30_out (Add)	(None, 16, 16, 1024)	0	conv4_block29_out[0][0] conv4_block30_3_conv[0][0]
conv4_block31_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block30_out[0][0]
conv4_block31_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block31_preact_bn[0][0]
conv4_block31_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block31_preact_relu[0][0]
conv4_block31_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block31_1_conv[0][0]
conv4_block31_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block31_1_bn[0][0]
conv4_block31_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block31_1_relu[0][0]
conv4_block31_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block31_2_pad[0][0]
conv4_block31_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block31_2_conv[0][0]
conv4_block31_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block31_2_bn[0][0]
conv4_block31_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block31_2_relu[0][0]
conv4_block31_out (Add)	(None, 16, 16, 1024)	0	conv4_block30_out[0][0] conv4_block31_3_conv[0][0]
conv4_block32_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block31_out[0][0]
conv4_block32_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block32_preact_bn[0][0]
conv4_block32_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block32_preact_relu[0][0]
conv4_block32_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block32_1_conv[0][0]
conv4_block32_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block32_1_bn[0][0]
conv4_block32_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block32_1_relu[0][0]
conv4_block32_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block32_2_pad[0][0]
conv4_block32_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block32_2_conv[0][0]
conv4_block32_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block32_2_bn[0][0]
conv4_block32_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block32_2_relu[0][0]
conv4_block32_out (Add)	(None, 16, 16, 1024)	0	conv4_block31_out[0][0] conv4_block32_3_conv[0][0]
conv4_block33_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block32_out[0][0]
conv4_block33_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block33_preact_bn[0][0]
conv4_block33_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block33_preact_relu[0][0]
conv4_block33_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block33_1_conv[0][0]
conv4_block33_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block33_1_bn[0][0]
conv4_block33_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block33_1_relu[0][0]
conv4_block33_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block33_2_pad[0][0]
conv4_block33_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block33_2_conv[0][0]
conv4_block33_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block33_2_bn[0][0]
conv4_block33_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block33_2_relu[0][0]
conv4_block33_out (Add)	(None, 16, 16, 1024)	0	conv4_block32_out[0][0] conv4_block33_3_conv[0][0]
conv4_block34_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block33_out[0][0]
conv4_block34_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block34_preact_bn[0][0]
conv4_block34_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block34_preact_relu[0][0]
conv4_block34_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block34_1_conv[0][0]
conv4_block34_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block34_1_bn[0][0]
conv4_block34_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block34_1_relu[0][0]
conv4_block34_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block34_2_pad[0][0]
conv4_block34_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block34_2_conv[0][0]
conv4_block34_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block34_2_bn[0][0]
conv4_block34_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block34_2_relu[0][0]
conv4_block34_out (Add)	(None, 16, 16, 1024)	0	conv4_block33_out[0][0] conv4_block34_3_conv[0][0]
conv4_block35_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block34_out[0][0]
conv4_block35_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block35_preact_bn[0][0]
conv4_block35_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block35_preact_relu[0][0]
conv4_block35_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block35_1_conv[0][0]
conv4_block35_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block35_1_bn[0][0]
conv4_block35_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block35_1_relu[0][0]
conv4_block35_2_conv (Conv2D)	(None, 16, 16, 256)	589824	conv4_block35_2_pad[0][0]
conv4_block35_2_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block35_2_conv[0][0]
conv4_block35_2_relu (Activatio	(None, 16, 16, 256)	0	conv4_block35_2_bn[0][0]
conv4_block35_3_conv (Conv2D)	(None, 16, 16, 1024)	263168	conv4_block35_2_relu[0][0]
conv4_block35_out (Add)	(None, 16, 16, 1024)	0	conv4_block34_out[0][0] conv4_block35_3_conv[0][0]
conv4_block36_preact_bn (BatchN	(None, 16, 16, 1024)	4096	conv4_block35_out[0][0]
conv4_block36_preact_relu (Acti	(None, 16, 16, 1024)	0	conv4_block36_preact_bn[0][0]
conv4_block36_1_conv (Conv2D)	(None, 16, 16, 256)	262144	conv4_block36_preact_relu[0][0]
conv4_block36_1_bn (BatchNormal	(None, 16, 16, 256)	1024	conv4_block36_1_conv[0][0]
conv4_block36_1_relu (Activatio	(None, 16, 16, 256)	0	conv4_block36_1_bn[0][0]
conv4_block36_2_pad (ZeroPaddin	(None, 18, 18, 256)	0	conv4_block36_1_relu[0][0]
conv4_block36_2_conv (Conv2D)	(None, 8, 8, 256)	589824	conv4_block36_2_pad[0][0]
conv4_block36_2_bn (BatchNormal	(None, 8, 8, 256)	1024	conv4_block36_2_conv[0][0]
conv4_block36_2_relu (Activatio	(None, 8, 8, 256)	0	conv4_block36_2_bn[0][0]
max_pooling2d_2 (MaxPooling2D)	(None, 8, 8, 1024)	0	conv4_block35_out[0][0]
conv4_block36_3_conv (Conv2D)	(None, 8, 8, 1024)	263168	conv4_block36_2_relu[0][0]
conv4_block36_out (Add)	(None, 8, 8, 1024)	0	max_pooling2d_2[0][0] conv4_block36_3_conv[0][0]
conv5_block1_preact_bn (BatchNo	(None, 8, 8, 1024)	4096	conv4_block36_out[0][0]
conv5_block1_preact_relu (Activ	(None, 8, 8, 1024)	0	conv5_block1_preact_bn[0][0]
conv5_block1_1_conv (Conv2D)	(None, 8, 8, 512)	524288	conv5_block1_preact_relu[0][0]
conv5_block1_1_bn (BatchNormali	(None, 8, 8, 512)	2048	conv5_block1_1_conv[0][0]
conv5_block1_1_relu (Activation	(None, 8, 8, 512)	0	conv5_block1_1_bn[0][0]
conv5_block1_2_pad (ZeroPadding	(None, 10, 10, 512)	0	conv5_block1_1_relu[0][0]
conv5_block1_2_conv (Conv2D)	(None, 8, 8, 512)	2359296	conv5_block1_2_pad[0][0]
conv5_block1_2_bn (BatchNormali	(None, 8, 8, 512)	2048	conv5_block1_2_conv[0][0]
conv5_block1_2_relu (Activation	(None, 8, 8, 512)	0	conv5_block1_2_bn[0][0]

```
____________________________________________________________________________________________________
conv5_block1_0_conv (Conv2D)    (None, 8, 8, 2048)   2099200     conv5_block1_preact_relu[0][0]
____________________________________________________________________________________________________
conv5_block1_3_conv (Conv2D)    (None, 8, 8, 2048)   1050624     conv5_block1_2_relu[0][0]
____________________________________________________________________________________________________
conv5_block1_out (Add)          (None, 8, 8, 2048)   0           conv5_block1_0_conv[0][0]
                                                                 conv5_block1_3_conv[0][0]
____________________________________________________________________________________________________
conv5_block2_preact_bn (BatchNo (None, 8, 8, 2048)   8192        conv5_block1_out[0][0]
____________________________________________________________________________________________________
conv5_block2_preact_relu (Activ (None, 8, 8, 2048)   0           conv5_block2_preact_bn[0][0]
____________________________________________________________________________________________________
conv5_block2_1_conv (Conv2D)    (None, 8, 8, 512)    1048576     conv5_block2_preact_relu[0][0]
____________________________________________________________________________________________________
conv5_block2_1_bn (BatchNormali (None, 8, 8, 512)    2048        conv5_block2_1_conv[0][0]
____________________________________________________________________________________________________
conv5_block2_1_relu (Activation (None, 8, 8, 512)    0           conv5_block2_1_bn[0][0]
____________________________________________________________________________________________________
conv5_block2_2_pad (ZeroPadding (None, 10, 10, 512)  0           conv5_block2_1_relu[0][0]
____________________________________________________________________________________________________
conv5_block2_2_conv (Conv2D)    (None, 8, 8, 512)    2359296     conv5_block2_2_pad[0][0]
____________________________________________________________________________________________________
conv5_block2_2_bn (BatchNormali (None, 8, 8, 512)    2048        conv5_block2_2_conv[0][0]
____________________________________________________________________________________________________
conv5_block2_2_relu (Activation (None, 8, 8, 512)    0           conv5_block2_2_bn[0][0]
____________________________________________________________________________________________________
conv5_block2_3_conv (Conv2D)    (None, 8, 8, 2048)   1050624     conv5_block2_2_relu[0][0]
____________________________________________________________________________________________________
conv5_block2_out (Add)          (None, 8, 8, 2048)   0           conv5_block1_out[0][0]
                                                                 conv5_block2_3_conv[0][0]
____________________________________________________________________________________________________
conv5_block3_preact_bn (BatchNo (None, 8, 8, 2048)   8192        conv5_block2_out[0][0]
____________________________________________________________________________________________________
conv5_block3_preact_relu (Activ (None, 8, 8, 2048)   0           conv5_block3_preact_bn[0][0]
____________________________________________________________________________________________________
conv5_block3_1_conv (Conv2D)    (None, 8, 8, 512)    1048576     conv5_block3_preact_relu[0][0]
____________________________________________________________________________________________________
conv5_block3_1_bn (BatchNormali (None, 8, 8, 512)    2048        conv5_block3_1_conv[0][0]
____________________________________________________________________________________________________
conv5_block3_1_relu (Activation (None, 8, 8, 512)    0           conv5_block3_1_bn[0][0]
____________________________________________________________________________________________________
conv5_block3_2_pad (ZeroPadding (None, 10, 10, 512)  0           conv5_block3_1_relu[0][0]
____________________________________________________________________________________________________
conv5_block3_2_conv (Conv2D)    (None, 8, 8, 512)    2359296     conv5_block3_2_pad[0][0]
____________________________________________________________________________________________________
conv5_block3_2_bn (BatchNormali (None, 8, 8, 512)    2048        conv5_block3_2_conv[0][0]
____________________________________________________________________________________________________
conv5_block3_2_relu (Activation (None, 8, 8, 512)    0           conv5_block3_2_bn[0][0]
____________________________________________________________________________________________________
conv5_block3_3_conv (Conv2D)    (None, 8, 8, 2048)   1050624     conv5_block3_2_relu[0][0]
____________________________________________________________________________________________________
conv5_block3_out (Add)          (None, 8, 8, 2048)   0           conv5_block2_out[0][0]
                                                                 conv5_block3_3_conv[0][0]
____________________________________________________________________________________________________
post_bn (BatchNormalization)    (None, 8, 8, 2048)   8192        conv5_block3_out[0][0]
____________________________________________________________________________________________________
post_relu (Activation)          (None, 8, 8, 2048)   0           post_bn[0][0]
____________________________________________________________________________________________________
flatten_1 (Flatten)             (None, 131072)       0           post_relu[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                 (None, 2)            262146      flatten_1[0][0]
====================================================================================================
Total params: 58,593,794
Trainable params: 262,146
Non-trainable params: 58,331,648
____________________________________________________________________________________________________
```

INCEPTIONV3

In [17]:

```
base_model = InceptionV3(input_shape=(256,256,3),include_top=False)
for layer in base_model.layers:
    layer.trainable=False

X=Flatten()(base_model.output)
X=Dense(units=2, activation='sigmoid')(X)

model_IV3 = Model(base_model.input, X)

model_IV3.compile(optimizer='adam',loss=keras.losses.binary_crossentropy,metrics=['accuracy'])
model_IV3.summary()
```

```
Model: "model_4"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_5 (InputLayer)            [(None, 256, 256, 3) 0
__________________________________________________________________________________________________
conv2d_94 (Conv2D)              (None, 127, 127, 32) 864         input_5[0][0]
__________________________________________________________________________________________________
batch_normalization_94 (BatchNo (None, 127, 127, 32) 96          conv2d_94[0][0]
__________________________________________________________________________________________________
activation_94 (Activation)      (None, 127, 127, 32) 0           batch_normalization_94[0][0]
__________________________________________________________________________________________________
conv2d_95 (Conv2D)              (None, 125, 125, 32) 9216        activation_94[0][0]
__________________________________________________________________________________________________
batch_normalization_95 (BatchNo (None, 125, 125, 32) 96          conv2d_95[0][0]
__________________________________________________________________________________________________
activation_95 (Activation)      (None, 125, 125, 32) 0           batch_normalization_95[0][0]
__________________________________________________________________________________________________
conv2d_96 (Conv2D)              (None, 125, 125, 64) 18432       activation_95[0][0]
__________________________________________________________________________________________________
batch_normalization_96 (BatchNo (None, 125, 125, 64) 192         conv2d_96[0][0]
__________________________________________________________________________________________________
activation_96 (Activation)      (None, 125, 125, 64) 0           batch_normalization_96[0][0]
__________________________________________________________________________________________________
max_pooling2d_7 (MaxPooling2D)  (None, 62, 62, 64)   0           activation_96[0][0]
__________________________________________________________________________________________________
conv2d_97 (Conv2D)              (None, 62, 62, 80)   5120        max_pooling2d_7[0][0]
__________________________________________________________________________________________________
batch_normalization_97 (BatchNo (None, 62, 62, 80)   240         conv2d_97[0][0]
__________________________________________________________________________________________________
activation_97 (Activation)      (None, 62, 62, 80)   0           batch_normalization_97[0][0]
__________________________________________________________________________________________________
conv2d_98 (Conv2D)              (None, 60, 60, 192)  138240      activation_97[0][0]
__________________________________________________________________________________________________
batch_normalization_98 (BatchNo (None, 60, 60, 192)  576         conv2d_98[0][0]
__________________________________________________________________________________________________
activation_98 (Activation)      (None, 60, 60, 192)  0           batch_normalization_98[0][0]
__________________________________________________________________________________________________
max_pooling2d_8 (MaxPooling2D)  (None, 29, 29, 192)  0           activation_98[0][0]
__________________________________________________________________________________________________
conv2d_102 (Conv2D)             (None, 29, 29, 64)   12288       max_pooling2d_8[0][0]
__________________________________________________________________________________________________
batch_normalization_102 (BatchN (None, 29, 29, 64)   192         conv2d_102[0][0]
__________________________________________________________________________________________________
activation_102 (Activation)     (None, 29, 29, 64)   0           batch_normalization_102[0][0]
__________________________________________________________________________________________________
conv2d_100 (Conv2D)             (None, 29, 29, 48)   9216        max_pooling2d_8[0][0]
__________________________________________________________________________________________________
conv2d_103 (Conv2D)             (None, 29, 29, 96)   55296       activation_102[0][0]
__________________________________________________________________________________________________
batch_normalization_100 (BatchN (None, 29, 29, 48)   144         conv2d_100[0][0]
__________________________________________________________________________________________________
batch_normalization_103 (BatchN (None, 29, 29, 96)   288         conv2d_103[0][0]
__________________________________________________________________________________________________
activation_100 (Activation)     (None, 29, 29, 48)   0           batch_normalization_100[0][0]
__________________________________________________________________________________________________
activation_103 (Activation)     (None, 29, 29, 96)   0           batch_normalization_103[0][0]
__________________________________________________________________________________________________
```

average_pooling2d_9 (AveragePoo	(None, 29, 29, 192)	0	max_pooling2d_8[0][0]
conv2d_99 (Conv2D)	(None, 29, 29, 64)	12288	max_pooling2d_8[0][0]
conv2d_101 (Conv2D)	(None, 29, 29, 64)	76800	activation_100[0][0]
conv2d_104 (Conv2D)	(None, 29, 29, 96)	82944	activation_103[0][0]
conv2d_105 (Conv2D)	(None, 29, 29, 32)	6144	average_pooling2d_9[0][0]
batch_normalization_99 (BatchNo	(None, 29, 29, 64)	192	conv2d_99[0][0]
batch_normalization_101 (BatchN	(None, 29, 29, 64)	192	conv2d_101[0][0]
batch_normalization_104 (BatchN	(None, 29, 29, 96)	288	conv2d_104[0][0]
batch_normalization_105 (BatchN	(None, 29, 29, 32)	96	conv2d_105[0][0]
activation_99 (Activation)	(None, 29, 29, 64)	0	batch_normalization_99[0][0]
activation_101 (Activation)	(None, 29, 29, 64)	0	batch_normalization_101[0][0]
activation_104 (Activation)	(None, 29, 29, 96)	0	batch_normalization_104[0][0]
activation_105 (Activation)	(None, 29, 29, 32)	0	batch_normalization_105[0][0]
mixed0 (Concatenate)	(None, 29, 29, 256)	0	activation_99[0][0] activation_101[0][0] activation_104[0][0] activation_105[0][0]
conv2d_109 (Conv2D)	(None, 29, 29, 64)	16384	mixed0[0][0]
batch_normalization_109 (BatchN	(None, 29, 29, 64)	192	conv2d_109[0][0]
activation_109 (Activation)	(None, 29, 29, 64)	0	batch_normalization_109[0][0]
conv2d_107 (Conv2D)	(None, 29, 29, 48)	12288	mixed0[0][0]
conv2d_110 (Conv2D)	(None, 29, 29, 96)	55296	activation_109[0][0]
batch_normalization_107 (BatchN	(None, 29, 29, 48)	144	conv2d_107[0][0]
batch_normalization_110 (BatchN	(None, 29, 29, 96)	288	conv2d_110[0][0]
activation_107 (Activation)	(None, 29, 29, 48)	0	batch_normalization_107[0][0]
activation_110 (Activation)	(None, 29, 29, 96)	0	batch_normalization_110[0][0]
average_pooling2d_10 (AveragePo	(None, 29, 29, 256)	0	mixed0[0][0]
conv2d_106 (Conv2D)	(None, 29, 29, 64)	16384	mixed0[0][0]
conv2d_108 (Conv2D)	(None, 29, 29, 64)	76800	activation_107[0][0]
conv2d_111 (Conv2D)	(None, 29, 29, 96)	82944	activation_110[0][0]
conv2d_112 (Conv2D)	(None, 29, 29, 64)	16384	average_pooling2d_10[0][0]
batch_normalization_106 (BatchN	(None, 29, 29, 64)	192	conv2d_106[0][0]
batch_normalization_108 (BatchN	(None, 29, 29, 64)	192	conv2d_108[0][0]
batch_normalization_111 (BatchN	(None, 29, 29, 96)	288	conv2d_111[0][0]
batch_normalization_112 (BatchN	(None, 29, 29, 64)	192	conv2d_112[0][0]
activation_106 (Activation)	(None, 29, 29, 64)	0	batch_normalization_106[0][0]
activation_108 (Activation)	(None, 29, 29, 64)	0	batch_normalization_108[0][0]
activation_111 (Activation)	(None, 29, 29, 96)	0	batch_normalization_111[0][0]
activation_112 (Activation)	(None, 29, 29, 64)	0	batch_normalization_112[0][0]
mixed1 (Concatenate)	(None, 29, 29, 288)	0	activation_106[0][0] activation_108[0][0] activation_111[0][0] activation_112[0][0]
conv2d_116 (Conv2D)	(None, 29, 29, 64)	18432	mixed1[0][0]
batch_normalization_116 (BatchN	(None, 29, 29, 64)	192	conv2d_116[0][0]
activation_116 (Activation)	(None, 29, 29, 64)	0	batch_normalization_116[0][0]
conv2d_114 (Conv2D)	(None, 29, 29, 48)	13824	mixed1[0][0]
conv2d_117 (Conv2D)	(None, 29, 29, 96)	55296	activation_116[0][0]
batch_normalization_114 (BatchN	(None, 29, 29, 48)	144	conv2d_114[0][0]
batch_normalization_117 (BatchN	(None, 29, 29, 96)	288	conv2d_117[0][0]
activation_114 (Activation)	(None, 29, 29, 48)	0	batch_normalization_114[0][0]
activation_117 (Activation)	(None, 29, 29, 96)	0	batch_normalization_117[0][0]
average_pooling2d_11 (AveragePo	(None, 29, 29, 288)	0	mixed1[0][0]
conv2d_113 (Conv2D)	(None, 29, 29, 64)	18432	mixed1[0][0]
conv2d_115 (Conv2D)	(None, 29, 29, 64)	76800	activation_114[0][0]
conv2d_118 (Conv2D)	(None, 29, 29, 96)	82944	activation_117[0][0]
conv2d_119 (Conv2D)	(None, 29, 29, 64)	18432	average_pooling2d_11[0][0]
batch_normalization_113 (BatchN	(None, 29, 29, 64)	192	conv2d_113[0][0]
batch_normalization_115 (BatchN	(None, 29, 29, 64)	192	conv2d_115[0][0]
batch_normalization_118 (BatchN	(None, 29, 29, 96)	288	conv2d_118[0][0]
batch_normalization_119 (BatchN	(None, 29, 29, 64)	192	conv2d_119[0][0]
activation_113 (Activation)	(None, 29, 29, 64)	0	batch_normalization_113[0][0]
activation_115 (Activation)	(None, 29, 29, 64)	0	batch_normalization_115[0][0]
activation_118 (Activation)	(None, 29, 29, 96)	0	batch_normalization_118[0][0]
activation_119 (Activation)	(None, 29, 29, 64)	0	batch_normalization_119[0][0]
mixed2 (Concatenate)	(None, 29, 29, 288)	0	activation_113[0][0] activation_115[0][0] activation_118[0][0] activation_119[0][0]
conv2d_121 (Conv2D)	(None, 29, 29, 64)	18432	mixed2[0][0]
batch_normalization_121 (BatchN	(None, 29, 29, 64)	192	conv2d_121[0][0]
activation_121 (Activation)	(None, 29, 29, 64)	0	batch_normalization_121[0][0]
conv2d_122 (Conv2D)	(None, 29, 29, 96)	55296	activation_121[0][0]
batch_normalization_122 (BatchN	(None, 29, 29, 96)	288	conv2d_122[0][0]
activation_122 (Activation)	(None, 29, 29, 96)	0	batch_normalization_122[0][0]
conv2d_120 (Conv2D)	(None, 14, 14, 384)	995328	mixed2[0][0]
conv2d_123 (Conv2D)	(None, 14, 14, 96)	82944	activation_122[0][0]
batch_normalization_120 (BatchN	(None, 14, 14, 384)	1152	conv2d_120[0][0]
batch_normalization_123 (BatchN	(None, 14, 14, 96)	288	conv2d_123[0][0]
activation_120 (Activation)	(None, 14, 14, 384)	0	batch_normalization_120[0][0]
activation_123 (Activation)	(None, 14, 14, 96)	0	batch_normalization_123[0][0]
max_pooling2d_9 (MaxPooling2D)	(None, 14, 14, 288)	0	mixed2[0][0]
mixed3 (Concatenate)	(None, 14, 14, 768)	0	activation_120[0][0] activation_123[0][0] max_pooling2d_9[0][0]
conv2d_128 (Conv2D)	(None, 14, 14, 128)	98304	mixed3[0][0]
batch_normalization_128 (BatchN	(None, 14, 14, 128)	384	conv2d_128[0][0]
activation_128 (Activation)	(None, 14, 14, 128)	0	batch_normalization_128[0][0]
conv2d_129 (Conv2D)	(None, 14, 14, 128)	114688	activation_128[0][0]
batch_normalization_129 (BatchN	(None, 14, 14, 128)	384	conv2d_129[0][0]
activation_129 (Activation)	(None, 14, 14, 128)	0	batch_normalization_129[0][0]
conv2d_125 (Conv2D)	(None, 14, 14, 128)	98304	mixed3[0][0]
conv2d_130 (Conv2D)	(None, 14, 14, 128)	114688	activation_129[0][0]
batch_normalization_125 (BatchN	(None, 14, 14, 128)	384	conv2d_125[0][0]
batch_normalization_130 (BatchN	(None, 14, 14, 128)	384	conv2d_130[0][0]
activation_125 (Activation)	(None, 14, 14, 128)	0	batch_normalization_125[0][0]
activation_130 (Activation)	(None, 14, 14, 128)	0	batch_normalization_130[0][0]
conv2d_126 (Conv2D)	(None, 14, 14, 128)	114688	activation_125[0][0]
conv2d_131 (Conv2D)	(None, 14, 14, 128)	114688	activation_130[0][0]
batch_normalization_126 (BatchN	(None, 14, 14, 128)	384	conv2d_126[0][0]
batch_normalization_131 (BatchN	(None, 14, 14, 128)	384	conv2d_131[0][0]
activation_126 (Activation)	(None, 14, 14, 128)	0	batch_normalization_126[0][0]
activation_131 (Activation)	(None, 14, 14, 128)	0	batch_normalization_131[0][0]
average_pooling2d_12 (AveragePo	(None, 14, 14, 768)	0	mixed3[0][0]
conv2d_124 (Conv2D)	(None, 14, 14, 192)	147456	mixed3[0][0]
conv2d_127 (Conv2D)	(None, 14, 14, 192)	172032	activation_126[0][0]
conv2d_132 (Conv2D)	(None, 14, 14, 192)	172032	activation_131[0][0]
conv2d_133 (Conv2D)	(None, 14, 14, 192)	147456	average_pooling2d_12[0][0]
batch_normalization_124 (BatchN	(None, 14, 14, 192)	576	conv2d_124[0][0]
batch_normalization_127 (BatchN	(None, 14, 14, 192)	576	conv2d_127[0][0]
batch_normalization_132 (BatchN	(None, 14, 14, 192)	576	conv2d_132[0][0]
batch_normalization_133 (BatchN	(None, 14, 14, 192)	576	conv2d_133[0][0]
activation_124 (Activation)	(None, 14, 14, 192)	0	batch_normalization_124[0][0]
activation_127 (Activation)	(None, 14, 14, 192)	0	batch_normalization_127[0][0]
activation_132 (Activation)	(None, 14, 14, 192)	0	batch_normalization_132[0][0]
activation_133 (Activation)	(None, 14, 14, 192)	0	batch_normalization_133[0][0]
mixed4 (Concatenate)	(None, 14, 14, 768)	0	activation_124[0][0] activation_127[0][0] activation_132[0][0] activation_133[0][0]
conv2d_138 (Conv2D)	(None, 14, 14, 160)	122880	mixed4[0][0]
batch_normalization_138 (BatchN	(None, 14, 14, 160)	480	conv2d_138[0][0]
activation_138 (Activation)	(None, 14, 14, 160)	0	batch_normalization_138[0][0]
conv2d_139 (Conv2D)	(None, 14, 14, 160)	179200	activation_138[0][0]
batch_normalization_139 (BatchN	(None, 14, 14, 160)	480	conv2d_139[0][0]
activation_139 (Activation)	(None, 14, 14, 160)	0	batch_normalization_139[0][0]

conv2d_135 (Conv2D)	(None, 14, 14, 160)	122880	mixed4[0][0]
conv2d_140 (Conv2D)	(None, 14, 14, 160)	179200	activation_139[0][0]
batch_normalization_135 (BatchN	(None, 14, 14, 160)	480	conv2d_135[0][0]
batch_normalization_140 (BatchN	(None, 14, 14, 160)	480	conv2d_140[0][0]
activation_135 (Activation)	(None, 14, 14, 160)	0	batch_normalization_135[0][0]
activation_140 (Activation)	(None, 14, 14, 160)	0	batch_normalization_140[0][0]
conv2d_136 (Conv2D)	(None, 14, 14, 160)	179200	activation_135[0][0]
conv2d_141 (Conv2D)	(None, 14, 14, 160)	179200	activation_140[0][0]
batch_normalization_136 (BatchN	(None, 14, 14, 160)	480	conv2d_136[0][0]
batch_normalization_141 (BatchN	(None, 14, 14, 160)	480	conv2d_141[0][0]
activation_136 (Activation)	(None, 14, 14, 160)	0	batch_normalization_136[0][0]
activation_141 (Activation)	(None, 14, 14, 160)	0	batch_normalization_141[0][0]
average_pooling2d_13 (AveragePo	(None, 14, 14, 768)	0	mixed4[0][0]
conv2d_134 (Conv2D)	(None, 14, 14, 192)	147456	mixed4[0][0]
conv2d_137 (Conv2D)	(None, 14, 14, 192)	215040	activation_136[0][0]
conv2d_142 (Conv2D)	(None, 14, 14, 192)	215040	activation_141[0][0]
conv2d_143 (Conv2D)	(None, 14, 14, 192)	147456	average_pooling2d_13[0][0]
batch_normalization_134 (BatchN	(None, 14, 14, 192)	576	conv2d_134[0][0]
batch_normalization_137 (BatchN	(None, 14, 14, 192)	576	conv2d_137[0][0]
batch_normalization_142 (BatchN	(None, 14, 14, 192)	576	conv2d_142[0][0]
batch_normalization_143 (BatchN	(None, 14, 14, 192)	576	conv2d_143[0][0]
activation_134 (Activation)	(None, 14, 14, 192)	0	batch_normalization_134[0][0]
activation_137 (Activation)	(None, 14, 14, 192)	0	batch_normalization_137[0][0]
activation_142 (Activation)	(None, 14, 14, 192)	0	batch_normalization_142[0][0]
activation_143 (Activation)	(None, 14, 14, 192)	0	batch_normalization_143[0][0]
mixed5 (Concatenate)	(None, 14, 14, 768)	0	activation_134[0][0] activation_137[0][0] activation_142[0][0] activation_143[0][0]
conv2d_148 (Conv2D)	(None, 14, 14, 160)	122880	mixed5[0][0]
batch_normalization_148 (BatchN	(None, 14, 14, 160)	480	conv2d_148[0][0]
activation_148 (Activation)	(None, 14, 14, 160)	0	batch_normalization_148[0][0]
conv2d_149 (Conv2D)	(None, 14, 14, 160)	179200	activation_148[0][0]
batch_normalization_149 (BatchN	(None, 14, 14, 160)	480	conv2d_149[0][0]
activation_149 (Activation)	(None, 14, 14, 160)	0	batch_normalization_149[0][0]
conv2d_145 (Conv2D)	(None, 14, 14, 160)	122880	mixed5[0][0]
conv2d_150 (Conv2D)	(None, 14, 14, 160)	179200	activation_149[0][0]
batch_normalization_145 (BatchN	(None, 14, 14, 160)	480	conv2d_145[0][0]
batch_normalization_150 (BatchN	(None, 14, 14, 160)	480	conv2d_150[0][0]
activation_145 (Activation)	(None, 14, 14, 160)	0	batch_normalization_145[0][0]
activation_150 (Activation)	(None, 14, 14, 160)	0	batch_normalization_150[0][0]
conv2d_146 (Conv2D)	(None, 14, 14, 160)	179200	activation_145[0][0]
conv2d_151 (Conv2D)	(None, 14, 14, 160)	179200	activation_150[0][0]
batch_normalization_146 (BatchN	(None, 14, 14, 160)	480	conv2d_146[0][0]
batch_normalization_151 (BatchN	(None, 14, 14, 160)	480	conv2d_151[0][0]
activation_146 (Activation)	(None, 14, 14, 160)	0	batch_normalization_146[0][0]
activation_151 (Activation)	(None, 14, 14, 160)	0	batch_normalization_151[0][0]
average_pooling2d_14 (AveragePo	(None, 14, 14, 768)	0	mixed5[0][0]
conv2d_144 (Conv2D)	(None, 14, 14, 192)	147456	mixed5[0][0]
conv2d_147 (Conv2D)	(None, 14, 14, 192)	215040	activation_146[0][0]
conv2d_152 (Conv2D)	(None, 14, 14, 192)	215040	activation_151[0][0]
conv2d_153 (Conv2D)	(None, 14, 14, 192)	147456	average_pooling2d_14[0][0]
batch_normalization_144 (BatchN	(None, 14, 14, 192)	576	conv2d_144[0][0]
batch_normalization_147 (BatchN	(None, 14, 14, 192)	576	conv2d_147[0][0]
batch_normalization_152 (BatchN	(None, 14, 14, 192)	576	conv2d_152[0][0]
batch_normalization_153 (BatchN	(None, 14, 14, 192)	576	conv2d_153[0][0]
activation_144 (Activation)	(None, 14, 14, 192)	0	batch_normalization_144[0][0]
activation_147 (Activation)	(None, 14, 14, 192)	0	batch_normalization_147[0][0]
activation_152 (Activation)	(None, 14, 14, 192)	0	batch_normalization_152[0][0]
activation_153 (Activation)	(None, 14, 14, 192)	0	batch_normalization_153[0][0]
mixed6 (Concatenate)	(None, 14, 14, 768)	0	activation_144[0][0] activation_147[0][0] activation_152[0][0] activation_153[0][0]
conv2d_158 (Conv2D)	(None, 14, 14, 192)	147456	mixed6[0][0]
batch_normalization_158 (BatchN	(None, 14, 14, 192)	576	conv2d_158[0][0]
activation_158 (Activation)	(None, 14, 14, 192)	0	batch_normalization_158[0][0]
conv2d_159 (Conv2D)	(None, 14, 14, 192)	258048	activation_158[0][0]
batch_normalization_159 (BatchN	(None, 14, 14, 192)	576	conv2d_159[0][0]
activation_159 (Activation)	(None, 14, 14, 192)	0	batch_normalization_159[0][0]
conv2d_155 (Conv2D)	(None, 14, 14, 192)	147456	mixed6[0][0]
conv2d_160 (Conv2D)	(None, 14, 14, 192)	258048	activation_159[0][0]
batch_normalization_155 (BatchN	(None, 14, 14, 192)	576	conv2d_155[0][0]
batch_normalization_160 (BatchN	(None, 14, 14, 192)	576	conv2d_160[0][0]
activation_155 (Activation)	(None, 14, 14, 192)	0	batch_normalization_155[0][0]
activation_160 (Activation)	(None, 14, 14, 192)	0	batch_normalization_160[0][0]
conv2d_156 (Conv2D)	(None, 14, 14, 192)	258048	activation_155[0][0]
conv2d_161 (Conv2D)	(None, 14, 14, 192)	258048	activation_160[0][0]
batch_normalization_156 (BatchN	(None, 14, 14, 192)	576	conv2d_156[0][0]
batch_normalization_161 (BatchN	(None, 14, 14, 192)	576	conv2d_161[0][0]
activation_156 (Activation)	(None, 14, 14, 192)	0	batch_normalization_156[0][0]
activation_161 (Activation)	(None, 14, 14, 192)	0	batch_normalization_161[0][0]
average_pooling2d_15 (AveragePo	(None, 14, 14, 768)	0	mixed6[0][0]
conv2d_154 (Conv2D)	(None, 14, 14, 192)	147456	mixed6[0][0]
conv2d_157 (Conv2D)	(None, 14, 14, 192)	258048	activation_156[0][0]
conv2d_162 (Conv2D)	(None, 14, 14, 192)	258048	activation_161[0][0]
conv2d_163 (Conv2D)	(None, 14, 14, 192)	147456	average_pooling2d_15[0][0]
batch_normalization_154 (BatchN	(None, 14, 14, 192)	576	conv2d_154[0][0]
batch_normalization_157 (BatchN	(None, 14, 14, 192)	576	conv2d_157[0][0]
batch_normalization_162 (BatchN	(None, 14, 14, 192)	576	conv2d_162[0][0]
batch_normalization_163 (BatchN	(None, 14, 14, 192)	576	conv2d_163[0][0]
activation_154 (Activation)	(None, 14, 14, 192)	0	batch_normalization_154[0][0]
activation_157 (Activation)	(None, 14, 14, 192)	0	batch_normalization_157[0][0]
activation_162 (Activation)	(None, 14, 14, 192)	0	batch_normalization_162[0][0]
activation_163 (Activation)	(None, 14, 14, 192)	0	batch_normalization_163[0][0]
mixed7 (Concatenate)	(None, 14, 14, 768)	0	activation_154[0][0] activation_157[0][0] activation_162[0][0] activation_163[0][0]
conv2d_166 (Conv2D)	(None, 14, 14, 192)	147456	mixed7[0][0]
batch_normalization_166 (BatchN	(None, 14, 14, 192)	576	conv2d_166[0][0]
activation_166 (Activation)	(None, 14, 14, 192)	0	batch_normalization_166[0][0]
conv2d_167 (Conv2D)	(None, 14, 14, 192)	258048	activation_166[0][0]
batch_normalization_167 (BatchN	(None, 14, 14, 192)	576	conv2d_167[0][0]
activation_167 (Activation)	(None, 14, 14, 192)	0	batch_normalization_167[0][0]
conv2d_164 (Conv2D)	(None, 14, 14, 192)	147456	mixed7[0][0]
conv2d_168 (Conv2D)	(None, 14, 14, 192)	258048	activation_167[0][0]
batch_normalization_164 (BatchN	(None, 14, 14, 192)	576	conv2d_164[0][0]
batch_normalization_168 (BatchN	(None, 14, 14, 192)	576	conv2d_168[0][0]
activation_164 (Activation)	(None, 14, 14, 192)	0	batch_normalization_164[0][0]
activation_168 (Activation)	(None, 14, 14, 192)	0	batch_normalization_168[0][0]
conv2d_165 (Conv2D)	(None, 6, 6, 320)	552960	activation_164[0][0]
conv2d_169 (Conv2D)	(None, 6, 6, 192)	331776	activation_168[0][0]
batch_normalization_165 (BatchN	(None, 6, 6, 320)	960	conv2d_165[0][0]
batch_normalization_169 (BatchN	(None, 6, 6, 192)	576	conv2d_169[0][0]
activation_165 (Activation)	(None, 6, 6, 320)	0	batch_normalization_165[0][0]
activation_169 (Activation)	(None, 6, 6, 192)	0	batch_normalization_169[0][0]
max_pooling2d_10 (MaxPooling2D)	(None, 6, 6, 768)	0	mixed7[0][0]
mixed8 (Concatenate)	(None, 6, 6, 1280)	0	activation_165[0][0] activation_169[0][0] max_pooling2d_10[0][0]
conv2d_174 (Conv2D)	(None, 6, 6, 448)	573440	mixed8[0][0]

batch_normalization_174 (BatchN	(None, 6, 6, 448)	1344	conv2d_174[0][0]
activation_174 (Activation)	(None, 6, 6, 448)	0	batch_normalization_174[0][0]
conv2d_171 (Conv2D)	(None, 6, 6, 384)	491520	mixed8[0][0]
conv2d_175 (Conv2D)	(None, 6, 6, 384)	1548288	activation_174[0][0]
batch_normalization_171 (BatchN	(None, 6, 6, 384)	1152	conv2d_171[0][0]
batch_normalization_175 (BatchN	(None, 6, 6, 384)	1152	conv2d_175[0][0]
activation_171 (Activation)	(None, 6, 6, 384)	0	batch_normalization_171[0][0]
activation_175 (Activation)	(None, 6, 6, 384)	0	batch_normalization_175[0][0]
conv2d_172 (Conv2D)	(None, 6, 6, 384)	442368	activation_171[0][0]
conv2d_173 (Conv2D)	(None, 6, 6, 384)	442368	activation_171[0][0]
conv2d_176 (Conv2D)	(None, 6, 6, 384)	442368	activation_175[0][0]
conv2d_177 (Conv2D)	(None, 6, 6, 384)	442368	activation_175[0][0]
average_pooling2d_16 (AveragePo	(None, 6, 6, 1280)	0	mixed8[0][0]
conv2d_170 (Conv2D)	(None, 6, 6, 320)	409600	mixed8[0][0]
batch_normalization_172 (BatchN	(None, 6, 6, 384)	1152	conv2d_172[0][0]
batch_normalization_173 (BatchN	(None, 6, 6, 384)	1152	conv2d_173[0][0]
batch_normalization_176 (BatchN	(None, 6, 6, 384)	1152	conv2d_176[0][0]
batch_normalization_177 (BatchN	(None, 6, 6, 384)	1152	conv2d_177[0][0]
conv2d_178 (Conv2D)	(None, 6, 6, 192)	245760	average_pooling2d_16[0][0]
batch_normalization_170 (BatchN	(None, 6, 6, 320)	960	conv2d_170[0][0]
activation_172 (Activation)	(None, 6, 6, 384)	0	batch_normalization_172[0][0]
activation_173 (Activation)	(None, 6, 6, 384)	0	batch_normalization_173[0][0]
activation_176 (Activation)	(None, 6, 6, 384)	0	batch_normalization_176[0][0]
activation_177 (Activation)	(None, 6, 6, 384)	0	batch_normalization_177[0][0]
batch_normalization_178 (BatchN	(None, 6, 6, 192)	576	conv2d_178[0][0]
activation_170 (Activation)	(None, 6, 6, 320)	0	batch_normalization_170[0][0]
mixed9_0 (Concatenate)	(None, 6, 6, 768)	0	activation_172[0][0] activation_173[0][0]
concatenate_2 (Concatenate)	(None, 6, 6, 768)	0	activation_176[0][0] activation_177[0][0]
activation_178 (Activation)	(None, 6, 6, 192)	0	batch_normalization_178[0][0]
mixed9 (Concatenate)	(None, 6, 6, 2048)	0	activation_170[0][0] mixed9_0[0][0] concatenate_2[0][0] activation_178[0][0]
conv2d_183 (Conv2D)	(None, 6, 6, 448)	917504	mixed9[0][0]

conv2d_183 (Conv2D)	(None, 6, 6, 448)	917504	mixed9[0][0]
batch_normalization_183 (BatchN	(None, 6, 6, 448)	1344	conv2d_183[0][0]
activation_183 (Activation)	(None, 6, 6, 448)	0	batch_normalization_183[0][0]
conv2d_180 (Conv2D)	(None, 6, 6, 384)	786432	mixed9[0][0]
conv2d_184 (Conv2D)	(None, 6, 6, 384)	1548288	activation_183[0][0]
batch_normalization_180 (BatchN	(None, 6, 6, 384)	1152	conv2d_180[0][0]
batch_normalization_184 (BatchN	(None, 6, 6, 384)	1152	conv2d_184[0][0]
activation_180 (Activation)	(None, 6, 6, 384)	0	batch_normalization_180[0][0]
activation_184 (Activation)	(None, 6, 6, 384)	0	batch_normalization_184[0][0]
conv2d_181 (Conv2D)	(None, 6, 6, 384)	442368	activation_180[0][0]
conv2d_182 (Conv2D)	(None, 6, 6, 384)	442368	activation_180[0][0]
conv2d_185 (Conv2D)	(None, 6, 6, 384)	442368	activation_184[0][0]
conv2d_186 (Conv2D)	(None, 6, 6, 384)	442368	activation_184[0][0]
average_pooling2d_17 (AveragePo	(None, 6, 6, 2048)	0	mixed9[0][0]
conv2d_179 (Conv2D)	(None, 6, 6, 320)	655360	mixed9[0][0]
batch_normalization_181 (BatchN	(None, 6, 6, 384)	1152	conv2d_181[0][0]
batch_normalization_182 (BatchN	(None, 6, 6, 384)	1152	conv2d_182[0][0]
batch_normalization_185 (BatchN	(None, 6, 6, 384)	1152	conv2d_185[0][0]
batch_normalization_186 (BatchN	(None, 6, 6, 384)	1152	conv2d_186[0][0]
conv2d_187 (Conv2D)	(None, 6, 6, 192)	393216	average_pooling2d_17[0][0]
batch_normalization_179 (BatchN	(None, 6, 6, 320)	960	conv2d_179[0][0]
activation_181 (Activation)	(None, 6, 6, 384)	0	batch_normalization_181[0][0]
activation_182 (Activation)	(None, 6, 6, 384)	0	batch_normalization_182[0][0]
activation_185 (Activation)	(None, 6, 6, 384)	0	batch_normalization_185[0][0]
activation_186 (Activation)	(None, 6, 6, 384)	0	batch_normalization_186[0][0]
batch_normalization_187 (BatchN	(None, 6, 6, 192)	576	conv2d_187[0][0]
activation_179 (Activation)	(None, 6, 6, 320)	0	batch_normalization_179[0][0]
mixed9_1 (Concatenate)	(None, 6, 6, 768)	0	activation_181[0][0] activation_182[0][0]
concatenate_3 (Concatenate)	(None, 6, 6, 768)	0	activation_185[0][0] activation_186[0][0]
activation_187 (Activation)	(None, 6, 6, 192)	0	batch_normalization_187[0][0]
mixed10 (Concatenate)	(None, 6, 6, 2048)	0	activation_179[0][0] mixed9_1[0][0] concatenate_3[0][0] activation_187[0][0]

flatten_4 (Flatten)	(None, 73728)	0	mixed10[0][0]
dense_4 (Dense)	(None, 2)	147458	flatten_4[0][0]

Total params: 21,950,242
Trainable params: 147,458
Non-trainable params: 21,802,784

VGG

In [19]:

```
vgg = VGG16( input_shape=(256,256,3), include_top= False)

for layer in vgg.layers:
  layer.trainable = False

x = Flatten()(vgg.output)
x = Dense(units=2 , activation='sigmoid', name = 'predictions' )(x)

model_vgg = Model(vgg.input, x)

model_vgg.summary()
model_vgg.compile(optimizer='adam', loss = 'categorical_crossentropy', metrics=['accuracy'])
```

```
Model: "model_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_7 (InputLayer)         [(None, 256, 256, 3)]     0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 256, 256, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 256, 256, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 128, 128, 64)      0
_________________________________________________________________
block2_conv1 (Conv2D)        (None, 128, 128, 128)     73856
_________________________________________________________________
block2_conv2 (Conv2D)        (None, 128, 128, 128)     147584
_________________________________________________________________
block2_pool (MaxPooling2D)   (None, 64, 64, 128)       0
_________________________________________________________________
block3_conv1 (Conv2D)        (None, 64, 64, 256)       295168
_________________________________________________________________
block3_conv2 (Conv2D)        (None, 64, 64, 256)       590080
_________________________________________________________________
block3_conv3 (Conv2D)        (None, 64, 64, 256)       590080
_________________________________________________________________
block3_pool (MaxPooling2D)   (None, 32, 32, 256)       0
_________________________________________________________________
block4_conv1 (Conv2D)        (None, 32, 32, 512)       1180160
_________________________________________________________________
block4_conv2 (Conv2D)        (None, 32, 32, 512)       2359808
_________________________________________________________________
block4_conv3 (Conv2D)        (None, 32, 32, 512)       2359808
_________________________________________________________________
block4_pool (MaxPooling2D)   (None, 16, 16, 512)       0
_________________________________________________________________
block5_conv1 (Conv2D)        (None, 16, 16, 512)       2359808
_________________________________________________________________
block5_conv2 (Conv2D)        (None, 16, 16, 512)       2359808
_________________________________________________________________
block5_conv3 (Conv2D)        (None, 16, 16, 512)       2359808
_________________________________________________________________
block5_pool (MaxPooling2D)   (None, 8, 8, 512)         0
_________________________________________________________________
flatten_6 (Flatten)          (None, 32768)             0
_________________________________________________________________
predictions (Dense)          (None, 2)                 65538
=================================================================
Total params: 14,780,226
Trainable params: 65,538
Non-trainable params: 14,714,688
_________________________________________________________________
```

DATA PREPROCESSING

In [22]:
```
from keras.preprocessing.image import ImageDataGenerator

def preprocessingImages1(path):
    """
    input : path
    output : Preprocessed Images
    """
    image_data = ImageDataGenerator(featurewise_center=True,
                                    rotation_range=0.4,
                                    width_shift_range=0.3,
                                    zoom_range=0.4,
                                    shear_range=0.4,
                                    rescale=1/255,
                                    horizontal_flip= True)

    image = image_data.flow_from_directory(directory= path,
                                           target_size=(256,256),
                                           batch_size=32)
    return image

def preprocessingImages2(path):
    """
    input : path
    output : Preprocessed Images
    """
    image_data = ImageDataGenerator(rescale=1/255)
    image = image_data.flow_from_directory(directory= path,
                                           target_size=(256,256),
                                           batch_size=32)
    return image
```

In [23]:
```
path=r"./Data/Train"
train_data = preprocessingImages1(path)
```

```
Found 6494 images belonging to 2 classes.
```

In [24]:
```
path=r"./Data/Test"
test_data = preprocessingImages2(path)
```

```
Found 1382 images belonging to 2 classes.
```

In [25]:
```
path=r"./Data/Val"
val_data = preprocessingImages2(path)
```

```
Found 1382 images belonging to 2 classes.
```

In [26]:
```
train_data.class_indices
```

Out[26]:
```
{'Negative': 0, 'Positive': 1}
```

In [27]:
```
val_data.class_indices
```

Out[27]:
```
{'Negative': 0, 'Positive': 1}
```

EARLY STOPPING AND MODEL SAVEPOINT

In [28]:
```
from keras.callbacks import ModelCheckpoint

mcIV3= ModelCheckpoint(filepath="./Models/c19modelIV3.hdf5", monitor="val_accuracy", verbose=1, save_best_only= True)
cbIV3=[mcIV3]

mcVGG= ModelCheckpoint(filepath="./Models/c19modelVGG.hdf5", monitor="val_accuracy", verbose=1, save_best_only= True)
cbVGG=[mcVGG]

mcDN= ModelCheckpoint(filepath="./Models/c19modelDN.hdf5", monitor="val_accuracy", verbose=1, save_best_only= True)
cbDN=[mcDN]

mcRN= ModelCheckpoint(filepath="./Models/c19modelRN.hdf5", monitor="val_accuracy", verbose=1, save_best_only= True)
cbRN=[mcRN]
```

In [29]:
```
his_DN = model_DN.fit_generator(train_data, steps_per_epoch= 10, epochs= 30, validation_data= val_data , validation_steps= 8, callbacks=cbDN)
```

```
Epoch 1/30

2022-05-15 10:42:14.170509: I tensorflow/stream_executor/cuda/cuda_dnn.cc:369] Loaded cuDNN version 8005

10/10 [==============================] - 35s 2s/step - loss: 3.9298 - accuracy: 0.8375 - val_loss: 5.4282 - val_accuracy: 0.8945

Epoch 00001: val_accuracy improved from -inf to 0.89453, saving model to ./Models/c19modelDN.hdf5
Epoch 2/30
10/10 [==============================] - 16s 2s/step - loss: 2.1747 - accuracy: 0.9156 - val_loss: 1.0778 - val_accuracy: 0.9141

Epoch 00002: val_accuracy improved from 0.89453 to 0.91406, saving model to ./Models/c19modelDN.hdf5
Epoch 3/30
10/10 [==============================] - 19s 2s/step - loss: 1.3387 - accuracy: 0.8813 - val_loss: 0.8289 - val_accuracy: 0.9492

Epoch 00003: val_accuracy improved from 0.91406 to 0.94922, saving model to ./Models/c19modelDN.hdf5
Epoch 4/30
10/10 [==============================] - 17s 2s/step - loss: 1.0733 - accuracy: 0.9094 - val_loss: 0.3868 - val_accuracy: 0.9453

Epoch 00004: val_accuracy did not improve from 0.94922
Epoch 5/30
10/10 [==============================] - 17s 2s/step - loss: 0.5212 - accuracy: 0.9281 - val_loss: 0.4655 - val_accuracy: 0.9336

Epoch 00005: val_accuracy did not improve from 0.94922
Epoch 6/30
10/10 [==============================] - 19s 2s/step - loss: 0.2540 - accuracy: 0.9344 - val_loss: 0.6485 - val_accuracy: 0.9258

Epoch 00006: val_accuracy did not improve from 0.94922
Epoch 7/30
10/10 [==============================] - 17s 2s/step - loss: 0.3804 - accuracy: 0.9156 - val_loss: 0.2903 - val_accuracy: 0.9492

Epoch 00007: val_accuracy did not improve from 0.94922
Epoch 8/30
10/10 [==============================] - 17s 2s/step - loss: 0.3649 - accuracy: 0.9250 - val_loss: 0.2917 - val_accuracy: 0.9375

Epoch 00008: val_accuracy did not improve from 0.94922
Epoch 9/30
10/10 [==============================] - 18s 2s/step - loss: 0.4118 - accuracy: 0.9187 - val_loss: 0.2389 - val_accuracy: 0.9375

Epoch 00009: val_accuracy did not improve from 0.94922
Epoch 10/30
10/10 [==============================] - 17s 2s/step - loss: 0.1774 - accuracy: 0.9625 - val_loss: 0.1877 - val_accuracy: 0.9570

Epoch 00010: val_accuracy improved from 0.94922 to 0.95703, saving model to ./Models/c19modelDN.hdf5
Epoch 11/30
10/10 [==============================] - 16s 2s/step - loss: 0.1157 - accuracy: 0.9719 - val_loss: 0.1641 - val_accuracy: 0.9727
Epoch 00011: val_accuracy improved from 0.95703 to 0.97266, saving model to ./Models/c19modelDN.hdf5
Epoch 12/30
10/10 [==============================] - 18s 2s/step - loss: 0.3210 - accuracy: 0.9312 - val_loss: 0.1836 - val_accuracy: 0.9648

Epoch 00012: val_accuracy did not improve from 0.97266
Epoch 13/30
10/10 [==============================] - 17s 2s/step - loss: 0.2999 - accuracy: 0.9375 - val_loss: 0.1099 - val_accuracy: 0.9688

Epoch 00013: val_accuracy did not improve from 0.97266
Epoch 14/30
10/10 [==============================] - 17s 2s/step - loss: 0.1283 - accuracy: 0.9750 - val_loss: 0.3578 - val_accuracy: 0.9336

Epoch 00014: val_accuracy did not improve from 0.97266
Epoch 15/30
10/10 [==============================] - 18s 2s/step - loss: 0.1240 - accuracy: 0.9594 - val_loss: 0.1875 - val_accuracy: 0.9375

Epoch 00015: val_accuracy did not improve from 0.97266
Epoch 16/30
10/10 [==============================] - 18s 2s/step - loss: 0.2311 - accuracy: 0.9531 - val_loss: 0.2376 - val_accuracy: 0.9375

Epoch 00016: val_accuracy did not improve from 0.97266
Epoch 17/30
10/10 [==============================] - 17s 2s/step - loss: 0.2670 - accuracy: 0.9438 - val_loss: 0.1191 - val_accuracy: 0.9688

Epoch 00017: val_accuracy did not improve from 0.97266
Epoch 18/30
10/10 [==============================] - 17s 2s/step - loss: 0.2479 - accuracy: 0.9375 - val_loss: 0.1932 - val_accuracy: 0.9570

Epoch 00018: val_accuracy did not improve from 0.97266
Epoch 19/30
10/10 [==============================] - 17s 2s/step - loss: 0.1996 - accuracy: 0.9719 - val_loss: 0.1984 - val_accuracy: 0.9492

Epoch 00019: val_accuracy did not improve from 0.97266
Epoch 20/30
10/10 [==============================] - 16s 2s/step - loss: 0.1070 - accuracy: 0.9719 - val_loss: 0.3747 - val_accuracy: 0.9531

Epoch 00020: val_accuracy did not improve from 0.97266
Epoch 21/30
10/10 [==============================] - 17s 2s/step - loss: 0.2181 - accuracy: 0.9688 - val_loss: 0.2984 - val_accuracy: 0.9492

Epoch 00021: val_accuracy did not improve from 0.97266
Epoch 22/30
10/10 [==============================] - 18s 2s/step - loss: 0.1550 - accuracy: 0.9719 - val_loss: 0.2214 - val_accuracy: 0.9492

Epoch 00022: val_accuracy did not improve from 0.97266
Epoch 23/30
10/10 [==============================] - 15s 2s/step - loss: 0.2159 - accuracy: 0.9656 - val_loss: 0.1799 - val_accuracy: 0.9648

Epoch 00023: val_accuracy did not improve from 0.97266
Epoch 24/30
10/10 [==============================] - 18s 2s/step - loss: 0.4034 - accuracy: 0.9156 - val_loss: 0.4266 - val_accuracy: 0.9492

Epoch 00024: val_accuracy did not improve from 0.97266
Epoch 25/30
10/10 [==============================] - 18s 2s/step - loss: 0.3725 - accuracy: 0.9434 - val_loss: 0.2296 - val_accuracy: 0.9414

Epoch 00025: val_accuracy did not improve from 0.97266
Epoch 26/30
10/10 [==============================] - 16s 2s/step - loss: 0.4250 - accuracy: 0.9469 - val_loss: 0.3048 - val_accuracy: 0.9609

Epoch 00026: val_accuracy did not improve from 0.97266
Epoch 27/30
10/10 [==============================] - 16s 2s/step - loss: 0.3625 - accuracy: 0.9406 - val_loss: 0.6556 - val_accuracy: 0.9414

Epoch 00027: val_accuracy did not improve from 0.97266
Epoch 28/30
10/10 [==============================] - 17s 2s/step - loss: 0.3856 - accuracy: 0.9594 - val_loss: 0.1827 - val_accuracy: 0.9609

Epoch 00028: val_accuracy did not improve from 0.97266
Epoch 29/30
10/10 [==============================] - 18s 2s/step - loss: 0.3527 - accuracy: 0.9531 - val_loss: 0.1684 - val_accuracy: 0.9766

Epoch 00029: val_accuracy improved from 0.97266 to 0.97656, saving model to ./Models/c19modelDN.hdf5
Epoch 30/30
10/10 [==============================] - 17s 2s/step - loss: 0.4660 - accuracy: 0.9312 - val_loss: 0.4986 - val_accuracy: 0.9414

Epoch 00030: val_accuracy did not improve from 0.97656
```

In [30]:
```
his_RN = model_RN.fit_generator(train_data, steps_per_epoch= 10, epochs= 30, validation_data= val_data , validation_steps= 8, callbacks=cbRN)
```

```
Epoch 1/30
10/10 [==============================] - 30s 2s/step - loss: 3.6987 - accuracy: 0.8313 - val_loss: 3.2861 - val_accuracy:
0.9102

Epoch 00001: val_accuracy improved from -inf to 0.91016, saving model to ./Models/c19modelRN.hdf5
Epoch 2/30
10/10 [==============================] - 19s 2s/step - loss: 1.7299 - accuracy: 0.8875 - val_loss: 1.0197 - val_accuracy:
0.9062

Epoch 00002: val_accuracy did not improve from 0.91016
Epoch 3/30
10/10 [==============================] - 18s 2s/step - loss: 1.0902 - accuracy: 0.9088 - val_loss: 2.4828 - val_accuracy:
0.9258

Epoch 00003: val_accuracy improved from 0.91016 to 0.92578, saving model to ./Models/c19modelRN.hdf5
Epoch 4/30
10/10 [==============================] - 18s 2s/step - loss: 1.5829 - accuracy: 0.9062 - val_loss: 0.9067 - val_accuracy:
0.9336

Epoch 00004: val_accuracy improved from 0.92578 to 0.93359, saving model to ./Models/c19modelRN.hdf5
Epoch 5/30
10/10 [==============================] - 20s 2s/step - loss: 0.6571 - accuracy: 0.9500 - val_loss: 0.5907 - val_accuracy:
0.9648

Epoch 00005: val_accuracy improved from 0.93359 to 0.96484, saving model to ./Models/c19modelRN.hdf5
Epoch 6/30
10/10 [==============================] - 17s 2s/step - loss: 0.9051 - accuracy: 0.9406 - val_loss: 1.3489 - val_accuracy:
0.9492

Epoch 00006: val_accuracy did not improve from 0.96484
Epoch 7/30
10/10 [==============================] - 17s 2s/step - loss: 1.2707 - accuracy: 0.9187 - val_loss: 0.7026 - val_accuracy:
0.9531

Epoch 00007: val_accuracy did not improve from 0.96484
Epoch 8/30
10/10 [==============================] - 18s 2s/step - loss: 0.3832 - accuracy: 0.9625 - val_loss: 0.4574 - val_accuracy:
0.9727

Epoch 00008: val_accuracy improved from 0.96484 to 0.97266, saving model to ./Models/c19modelRN.hdf5
Epoch 9/30
10/10 [==============================] - 18s 2s/step - loss: 0.4758 - accuracy: 0.9500 - val_loss: 0.4561 - val_accuracy:
0.9688

Epoch 00009: val_accuracy did not improve from 0.97266
Epoch 10/30
10/10 [==============================] - 16s 2s/step - loss: 0.4934 - accuracy: 0.9563 - val_loss: 0.5922 - val_accuracy:
0.9414

Epoch 00010: val_accuracy did not improve from 0.97266
Epoch 11/30
10/10 [==============================] - 18s 2s/step - loss: 0.6828 - accuracy: 0.9344 - val_loss: 0.5274 - val_accuracy:
0.9531

Epoch 00011: val_accuracy did not improve from 0.97266
Epoch 12/30
10/10 [==============================] - 19s 2s/step - loss: 0.4201 - accuracy: 0.9625 - val_loss: 0.4361 - val_accuracy:
0.9492

Epoch 00012: val_accuracy did not improve from 0.97266
Epoch 13/30
10/10 [==============================] - 17s 2s/step - loss: 0.6606 - accuracy: 0.9500 - val_loss: 0.4689 - val_accuracy:
0.9648

Epoch 00013: val_accuracy did not improve from 0.97266
Epoch 14/30
10/10 [==============================] - 19s 2s/step - loss: 0.3701 - accuracy: 0.9594 - val_loss: 0.8689 - val_accuracy:
0.9453

Epoch 00014: val_accuracy did not improve from 0.97266
Epoch 15/30
10/10 [==============================] - 18s 2s/step - loss: 0.6574 - accuracy: 0.9406 - val_loss: 0.3180 - val_accuracy:
0.9492

Epoch 00015: val_accuracy did not improve from 0.97266
Epoch 16/30
10/10 [==============================] - 17s 2s/step - loss: 0.6675 - accuracy: 0.9250 - val_loss: 0.6387 - val_accuracy:
0.9531

Epoch 00016: val_accuracy did not improve from 0.97266
Epoch 17/30
10/10 [==============================] - 17s 2s/step - loss: 0.7458 - accuracy: 0.9406 - val_loss: 0.2718 - val_accuracy:
0.9805

Epoch 00017: val_accuracy improved from 0.97266 to 0.98047, saving model to ./Models/c19modelRN.hdf5
Epoch 18/30
10/10 [==============================] - 19s 2s/step - loss: 0.5741 - accuracy: 0.9406 - val_loss: 0.3120 - val_accuracy:
0.9766

Epoch 00018: val_accuracy did not improve from 0.98047
Epoch 19/30
10/10 [==============================] - 17s 2s/step - loss: 0.5441 - accuracy: 0.9625 - val_loss: 0.2799 - val_accuracy:
0.9727

Epoch 00019: val_accuracy did not improve from 0.98047
Epoch 20/30
10/10 [==============================] - 18s 2s/step - loss: 0.3996 - accuracy: 0.9656 - val_loss: 0.3957 - val_accuracy:
0.9609

Epoch 00020: val_accuracy did not improve from 0.98047
Epoch 21/30
10/10 [==============================] - 20s 2s/step - loss: 0.6615 - accuracy: 0.9312 - val_loss: 0.3549 - val_accuracy:
0.9727

Epoch 00021: val_accuracy did not improve from 0.98047
Epoch 22/30
10/10 [==============================] - 17s 2s/step - loss: 0.3964 - accuracy: 0.9531 - val_loss: 0.6186 - val_accuracy:
0.9609

Epoch 00022: val_accuracy did not improve from 0.98047
Epoch 23/30
10/10 [==============================] - 16s 2s/step - loss: 0.4692 - accuracy: 0.9656 - val_loss: 0.1877 - val_accuracy:
0.9766

Epoch 00023: val_accuracy did not improve from 0.98047
Epoch 24/30
10/10 [==============================] - 18s 2s/step - loss: 0.7211 - accuracy: 0.9500 - val_loss: 0.5863 - val_accuracy:
0.9570

Epoch 00024: val_accuracy did not improve from 0.98047
Epoch 25/30
10/10 [==============================] - 17s 2s/step - loss: 0.6000 - accuracy: 0.9500 - val_loss: 0.3008 - val_accuracy:
0.9688

Epoch 00025: val_accuracy did not improve from 0.98047
Epoch 26/30
10/10 [==============================] - 17s 2s/step - loss: 0.4178 - accuracy: 0.9719 - val_loss: 0.6580 - val_accuracy:
0.9414

Epoch 00026: val_accuracy did not improve from 0.98047
Epoch 27/30
10/10 [==============================] - 18s 2s/step - loss: 0.6769 - accuracy: 0.9531 - val_loss: 0.7943 - val_accuracy:
0.9258

Epoch 00027: val_accuracy did not improve from 0.98047
Epoch 28/30
10/10 [==============================] - 18s 2s/step - loss: 0.7468 - accuracy: 0.9375 - val_loss: 1.2832 - val_accuracy:
0.9258

Epoch 00028: val_accuracy did not improve from 0.98047
Epoch 29/30
10/10 [==============================] - 17s 2s/step - loss: 0.7104 - accuracy: 0.9438 - val_loss: 0.1885 - val_accuracy:
0.9805

Epoch 00029: val_accuracy did not improve from 0.98047
Epoch 30/30
10/10 [==============================] - 18s 2s/step - loss: 0.4340 - accuracy: 0.9438 - val_loss: 0.4037 - val_accuracy:
0.9648

Epoch 00030: val_accuracy did not improve from 0.98047
```

In [31]:

```
his_IV3 = model_IV3.fit_generator(train_data, steps_per_epoch= 10, epochs= 30, validation_data= val_data , validation_steps=
8, callbacks=cbIV3)
```

```
Epoch 1/30
10/10 [==============================] - 22s 2s/step - loss: 1.8880 - accuracy: 0.8938 - val_loss: 1.0709 - val_accuracy:
0.9102

Epoch 00001: val_accuracy improved from -inf to 0.91016, saving model to ./Models/c19modelIV3.hdf5
Epoch 2/30
10/10 [==============================] - 16s 2s/step - loss: 0.8539 - accuracy: 0.9187 - val_loss: 0.9784 - val_accuracy:
0.9219

Epoch 00002: val_accuracy improved from 0.91016 to 0.92188, saving model to ./Models/c19modelIV3.hdf5
Epoch 3/30
10/10 [==============================] - 19s 2s/step - loss: 0.9330 - accuracy: 0.9031 - val_loss: 0.4882 - val_accuracy:
0.9414

Epoch 00003: val_accuracy improved from 0.92188 to 0.94141, saving model to ./Models/c19modelIV3.hdf5
Epoch 4/30
10/10 [==============================] - 16s 2s/step - loss: 0.2700 - accuracy: 0.9531 - val_loss: 0.5412 - val_accuracy:
0.9180

Epoch 00004: val_accuracy did not improve from 0.94141
Epoch 5/30
10/10 [==============================] - 16s 2s/step - loss: 0.5815 - accuracy: 0.9250 - val_loss: 0.3547 - val_accuracy:
0.9492

Epoch 00005: val_accuracy improved from 0.94141 to 0.94922, saving model to ./Models/c19modelIV3.hdf5
Epoch 6/30
10/10 [==============================] - 17s 2s/step - loss: 0.4743 - accuracy: 0.9250 - val_loss: 0.2530 - val_accuracy:
0.9375

Epoch 00006: val_accuracy did not improve from 0.94922
Epoch 7/30
10/10 [==============================] - 17s 2s/step - loss: 0.2874 - accuracy: 0.9656 - val_loss: 0.4677 - val_accuracy:
0.9336

Epoch 00007: val_accuracy did not improve from 0.94922
Epoch 8/30
10/10 [==============================] - 16s 2s/step - loss: 0.4736 - accuracy: 0.9250 - val_loss: 0.4948 - val_accuracy:
0.9414

Epoch 00008: val_accuracy did not improve from 0.94922
Epoch 9/30
10/10 [==============================] - 18s 2s/step - loss: 0.3932 - accuracy: 0.9245 - val_loss: 0.8189 - val_accuracy:
0.8945

Epoch 00009: val_accuracy did not improve from 0.94922
Epoch 10/30
10/10 [==============================] - 16s 2s/step - loss: 0.5699 - accuracy: 0.9214 - val_loss: 0.9066 - val_accuracy:
0.9141

Epoch 00010: val_accuracy did not improve from 0.94922
Epoch 11/30
10/10 [==============================] - 16s 2s/step - loss: 0.4864 - accuracy: 0.9281 - val_loss: 0.5459 - val_accuracy:
0.9258
```

```
Epoch 00010: val_accuracy did not improve from 0.94922
Epoch 11/30
10/10 [==============================] - 16s 2s/step - loss: 0.4864 - accuracy: 0.9281 - val_loss: 0.5459 - val_accuracy:
0.9258

Epoch 00011: val_accuracy did not improve from 0.94922
Epoch 12/30
10/10 [==============================] - 18s 2s/step - loss: 0.5224 - accuracy: 0.9057 - val_loss: 0.4190 - val_accuracy:
0.9375

Epoch 00012: val_accuracy did not improve from 0.94922
Epoch 13/30
10/10 [==============================] - 16s 2s/step - loss: 0.5047 - accuracy: 0.9219 - val_loss: 0.4791 - val_accuracy:
0.9258

Epoch 00013: val_accuracy did not improve from 0.94922
Epoch 14/30
10/10 [==============================] - 16s 2s/step - loss: 0.2831 - accuracy: 0.9500 - val_loss: 0.2281 - val_accuracy:
0.9570

Epoch 00014: val_accuracy improved from 0.94922 to 0.95703, saving model to ./Models/c19modelIV3.hdf5
Epoch 15/30
10/10 [==============================] - 17s 2s/step - loss: 0.2382 - accuracy: 0.9625 - val_loss: 0.2991 - val_accuracy:
0.9688

Epoch 00015: val_accuracy improved from 0.95703 to 0.96875, saving model to ./Models/c19modelIV3.hdf5
Epoch 16/30
10/10 [==============================] - 17s 2s/step - loss: 0.4952 - accuracy: 0.9281 - val_loss: 0.2326 - val_accuracy:
0.9609
Epoch 00016: val_accuracy did not improve from 0.96875
Epoch 17/30
10/10 [==============================] - 15s 2s/step - loss: 0.2625 - accuracy: 0.9594 - val_loss: 0.3454 - val_accuracy:
0.9297

Epoch 00017: val_accuracy did not improve from 0.96875
Epoch 18/30
10/10 [==============================] - 17s 2s/step - loss: 0.3614 - accuracy: 0.9469 - val_loss: 0.1221 - val_accuracy:
0.9648

Epoch 00018: val_accuracy did not improve from 0.96875
Epoch 19/30
10/10 [==============================] - 18s 2s/step - loss: 0.2332 - accuracy: 0.9625 - val_loss: 0.6095 - val_accuracy:
0.8867

Epoch 00019: val_accuracy did not improve from 0.96875
Epoch 20/30
10/10 [==============================] - 16s 2s/step - loss: 0.7232 - accuracy: 0.8994 - val_loss: 0.3233 - val_accuracy:
0.9492

Epoch 00020: val_accuracy did not improve from 0.96875
Epoch 21/30
10/10 [==============================] - 18s 2s/step - loss: 0.3115 - accuracy: 0.9406 - val_loss: 1.1358 - val_accuracy:
0.9141

Epoch 00021: val_accuracy did not improve from 0.96875
Epoch 22/30
10/10 [==============================] - 17s 2s/step - loss: 0.7242 - accuracy: 0.9312 - val_loss: 0.7484 - val_accuracy:
0.9180

Epoch 00022: val_accuracy did not improve from 0.96875
Epoch 23/30
10/10 [==============================] - 15s 2s/step - loss: 0.5709 - accuracy: 0.9375 - val_loss: 0.3244 - val_accuracy:
0.9414

Epoch 00023: val_accuracy did not improve from 0.96875
Epoch 24/30
10/10 [==============================] - 18s 2s/step - loss: 0.4179 - accuracy: 0.9469 - val_loss: 0.2870 - val_accuracy:
0.9492

Epoch 00024: val_accuracy did not improve from 0.96875
Epoch 25/30
10/10 [==============================] - 17s 2s/step - loss: 0.3347 - accuracy: 0.9531 - val_loss: 0.2793 - val_accuracy:
0.9414

Epoch 00025: val_accuracy did not improve from 0.96875
Epoch 26/30
10/10 [==============================] - 17s 2s/step - loss: 0.4918 - accuracy: 0.9469 - val_loss: 0.1591 - val_accuracy:
0.9609

Epoch 00026: val_accuracy did not improve from 0.96875
Epoch 27/30
10/10 [==============================] - 17s 2s/step - loss: 0.6881 - accuracy: 0.9281 - val_loss: 0.7212 - val_accuracy:
0.9141

Epoch 00027: val_accuracy did not improve from 0.96875
Epoch 28/30
10/10 [==============================] - 18s 2s/step - loss: 0.2771 - accuracy: 0.9469 - val_loss: 0.3587 - val_accuracy:
0.9688

Epoch 00028: val_accuracy did not improve from 0.96875
Epoch 29/30
10/10 [==============================] - 17s 2s/step - loss: 0.5344 - accuracy: 0.9375 - val_loss: 0.4593 - val_accuracy:
0.9336

Epoch 00029: val_accuracy did not improve from 0.96875
Epoch 30/30
10/10 [==============================] - 17s 2s/step - loss: 0.4080 - accuracy: 0.9438 - val_loss: 0.2911 - val_accuracy:
0.9414

Epoch 00030: val_accuracy did not improve from 0.96875
```

In [32]:

```
his_VGG = model_vgg.fit_generator(train_data, steps_per_epoch= 10, epochs= 30, validation_data= val_data , validation_steps=
8, callbacks=cbVGG)
```

```
Epoch 1/30
10/10 [==============================] - 20s 2s/step - loss: 0.6549 - accuracy: 0.8875 - val_loss: 0.2493 - val_accuracy:
0.9062

Epoch 00001: val_accuracy improved from -inf to 0.90625, saving model to ./Models/c19modelVGG.hdf5
Epoch 2/30
10/10 [==============================] - 16s 2s/step - loss: 0.5449 - accuracy: 0.8969 - val_loss: 0.2059 - val_accuracy:
0.8984

Epoch 00002: val_accuracy did not improve from 0.90625
Epoch 3/30
10/10 [==============================] - 17s 2s/step - loss: 0.3450 - accuracy: 0.8719 - val_loss: 0.2812 - val_accuracy:
0.9258

Epoch 00003: val_accuracy improved from 0.90625 to 0.92578, saving model to ./Models/c19modelVGG.hdf5
Epoch 4/30
10/10 [==============================] - 18s 2s/step - loss: 0.2977 - accuracy: 0.9062 - val_loss: 0.3104 - val_accuracy:
0.9102

Epoch 00004: val_accuracy did not improve from 0.92578
Epoch 5/30
10/10 [==============================] - 18s 2s/step - loss: 0.2020 - accuracy: 0.9245 - val_loss: 0.1505 - val_accuracy:
0.9258

Epoch 00005: val_accuracy did not improve from 0.92578
Epoch 6/30
10/10 [==============================] - 16s 2s/step - loss: 0.1796 - accuracy: 0.9219 - val_loss: 0.1842 - val_accuracy:
0.9062

Epoch 00006: val_accuracy did not improve from 0.92578
Epoch 7/30
10/10 [==============================] - 17s 2s/step - loss: 0.1757 - accuracy: 0.9219 - val_loss: 0.1202 - val_accuracy:
0.9414
Epoch 00007: val_accuracy improved from 0.92578 to 0.94141, saving model to ./Models/c19modelVGG.hdf5
Epoch 8/30
10/10 [==============================] - 17s 2s/step - loss: 0.1984 - accuracy: 0.9000 - val_loss: 0.1391 - val_accuracy:
0.9336

Epoch 00008: val_accuracy did not improve from 0.94141
Epoch 9/30
10/10 [==============================] - 16s 2s/step - loss: 0.1882 - accuracy: 0.9125 - val_loss: 0.1906 - val_accuracy:
0.9023

Epoch 00009: val_accuracy did not improve from 0.94141
Epoch 10/30
10/10 [==============================] - 17s 2s/step - loss: 0.1594 - accuracy: 0.9406 - val_loss: 0.1890 - val_accuracy:
0.9141

Epoch 00010: val_accuracy did not improve from 0.94141
Epoch 11/30
10/10 [==============================] - 17s 2s/step - loss: 0.1227 - accuracy: 0.9500 - val_loss: 0.1350 - val_accuracy:
0.9531

Epoch 00013: val_accuracy did not improve from 0.95312
Epoch 14/30
10/10 [==============================] - 16s 2s/step - loss: 0.1604 - accuracy: 0.9219 - val_loss: 0.0794 - val_accuracy:
0.9727

Epoch 00014: val_accuracy improved from 0.95312 to 0.97266, saving model to ./Models/c19modelVGG.hdf5
Epoch 15/30
10/10 [==============================] - 16s 2s/step - loss: 0.1329 - accuracy: 0.9375 - val_loss: 0.0950 - val_accuracy:
0.9609

Epoch 00015: val_accuracy did not improve from 0.97266
Epoch 16/30
10/10 [==============================] - 16s 2s/step - loss: 0.1531 - accuracy: 0.9375 - val_loss: 0.1453 - val_accuracy:
0.9492

Epoch 00016: val_accuracy did not improve from 0.97266
Epoch 17/30
10/10 [==============================] - 19s 2s/step - loss: 0.1183 - accuracy: 0.9656 - val_loss: 0.1659 - val_accuracy:
0.9336

Epoch 00017: val_accuracy did not improve from 0.97266
Epoch 18/30
10/10 [==============================] - 16s 2s/step - loss: 0.1741 - accuracy: 0.9281 - val_loss: 0.1575 - val_accuracy:
0.9375

Epoch 00018: val_accuracy did not improve from 0.97266
Epoch 19/30
10/10 [==============================] - 17s 2s/step - loss: 0.1890 - accuracy: 0.9219 - val_loss: 0.1760 - val_accuracy:
0.9453

Epoch 00019: val_accuracy did not improve from 0.97266
Epoch 20/30
10/10 [==============================] - 18s 2s/step - loss: 0.1954 - accuracy: 0.9438 - val_loss: 0.1066 - val_accuracy:
0.9414
Epoch 00020: val_accuracy did not improve from 0.97266
Epoch 21/30
10/10 [==============================] - 15s 2s/step - loss: 0.1225 - accuracy: 0.9375 - val_loss: 0.1098 - val_accuracy:
0.9570

Epoch 00021: val_accuracy did not improve from 0.97266
Epoch 22/30
10/10 [==============================] - 17s 2s/step - loss: 0.1536 - accuracy: 0.9340 - val_loss: 0.0952 - val_accuracy:
0.9492

Epoch 00022: val_accuracy did not improve from 0.97266
Epoch 23/30
10/10 [==============================] - 19s 2s/step - loss: 0.1641 - accuracy: 0.9308 - val_loss: 0.1107 - val_accuracy:
0.9414

Epoch 00023: val_accuracy did not improve from 0.97266
Epoch 24/30
10/10 [==============================] - 16s 2s/step - loss: 0.1183 - accuracy: 0.9594 - val_loss: 0.1614 - val_accuracy:
0.9219

Epoch 00024: val_accuracy did not improve from 0.97266
Epoch 25/30
10/10 [==============================] - 17s 2s/step - loss: 0.0697 - accuracy: 0.9656 - val_loss: 0.1225 - val_accuracy:
0.9570

Epoch 00025: val_accuracy did not improve from 0.97266
Epoch 26/30
10/10 [==============================] - 17s 2s/step - loss: 0.1526 - accuracy: 0.9438 - val_loss: 0.0663 - val_accuracy:
0.9727

Epoch 00026: val_accuracy did not improve from 0.97266
Epoch 27/30
10/10 [==============================] - 16s 2s/step - loss: 0.1563 - accuracy: 0.9406 - val_loss: 0.1295 - val_accuracy:
0.9531
Epoch 00027: val_accuracy did not improve from 0.97266
Epoch 28/30
10/10 [==============================] - 17s 2s/step - loss: 0.1526 - accuracy: 0.9344 - val_loss: 0.0553 - val_accuracy:
0.9766

Epoch 00028: val_accuracy improved from 0.97266 to 0.97656, saving model to ./Models/c19modelVGG.hdf5
Epoch 29/30
10/10 [==============================] - 18s 2s/step - loss: 0.1830 - accuracy: 0.9344 - val_loss: 0.1701 - val_accuracy:
0.9336

Epoch 00029: val_accuracy did not improve from 0.97656
Epoch 30/30
10/10 [==============================] - 17s 2s/step - loss: 0.1847 - accuracy: 0.9375 - val_loss: 0.0735 - val_accuracy:
0.9609

Epoch 00030: val_accuracy did not improve from 0.97656
```

In [33]:
```
hDN = his_DN.history
hDN.keys()
```

Out[33]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

In [34]:
```
hRN = his_RN.history
hRN.keys()
```

Out[34]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

In [35]:
```
hIV3 = his_IV3.history
hIV3.keys()
```

Out[35]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

In [36]:
```
hVGG = his_VGG.history
hVGG.keys()
```

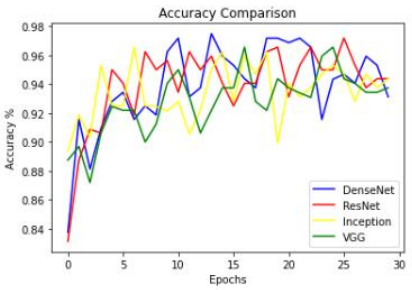
Out[36]:
```
dict_keys(['loss', 'accuracy', 'val_loss', 'val_accuracy'])
```

In [41]:
```
print("ACCURACY Comparison")

fig, ax = plt.subplots()
plt.plot(hDN['accuracy'], c='blue', label ='DenseNet')
plt.plot(hRN['accuracy'], c='red', label ='ResNet')
plt.plot(hIV3['accuracy'], c='yellow', label ='Inception')
plt.plot(hVGG['accuracy'], c='green', label ='VGG')
leg = ax.legend();
plt.xlabel('Epochs')
plt.ylabel('Accuracy %')
plt.title("Accuracy Comparison")
plt.show
```

```
ACCURACY Comparison
```

Out[41]:
```
<function matplotlib.pyplot.show(close=None, block=None)>
```



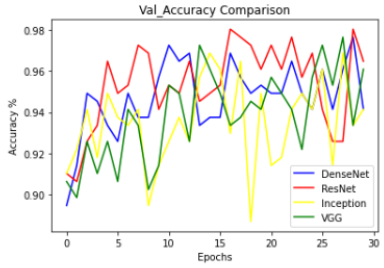
In [42]:
```
print("Val_ACCURACY Comparison")

fig, ax = plt.subplots()
plt.plot(hDN['val_accuracy'], c='blue', label ='DenseNet')
plt.plot(hRN['val_accuracy'], c='red', label ='ResNet')
plt.plot(hIV3['val_accuracy'], c='yellow', label ='Inception')
plt.plot(hVGG['val_accuracy'], c='green', label ='VGG')
leg = ax.legend();
plt.xlabel('Epochs')
plt.ylabel('Accuracy %')
plt.title("Val_Accuracy Comparison")
plt.show
```

Val_ACCURACY Comparison

Out[42]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [43]:

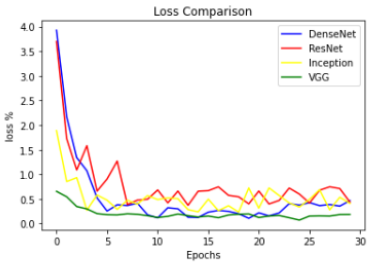
```
print("LOSS Comparison")

fig, ax = plt.subplots()
plt.plot(hDN['loss'], c='blue', label ='DenseNet')
plt.plot(hRN['loss'], c='red', label ='ResNet')
plt.plot(hIV3['loss'], c='yellow', label ='Inception')
plt.plot(hVGG['loss'], c='green', label ='VGG')
leg = ax.legend();
plt.title("Loss Comparison")
plt.xlabel('Epochs')
plt.ylabel('loss %')
plt.show
```

LOSS Comparison

Out[43]:

```
<function matplotlib.pyplot.show(close=None, block=None)>
```



In [44]:

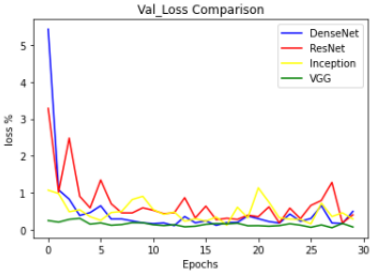
```
print("Val_LOSS Comparison")

fig, ax = plt.subplots()
plt.plot(hDN['val_loss'], c='blue', label ='DenseNet')
plt.plot(hRN['val_loss'], c='red', label ='ResNet')
plt.plot(hIV3['val_loss'], c='yellow', label ='Inception')
plt.plot(hVGG['val_loss'], c='green', label ='VGG')
leg = ax.legend();
plt.xlabel('Epochs')
plt.ylabel('loss %')
plt.title("Val_Loss Comparison")
plt.show
```

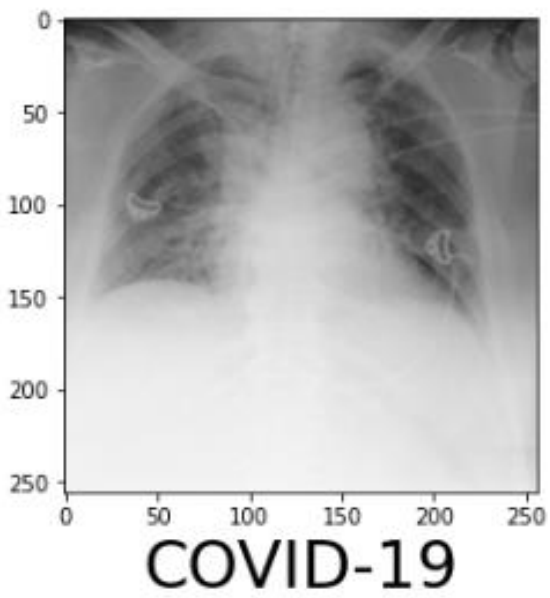
Val_LOSS Comparison

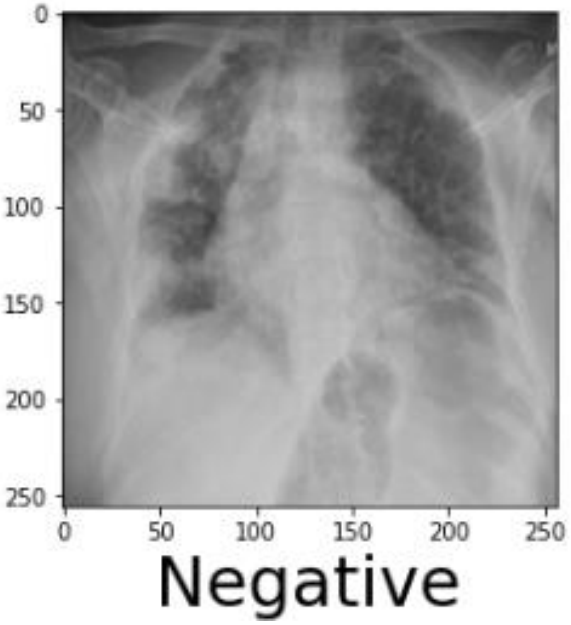
Out[44]:

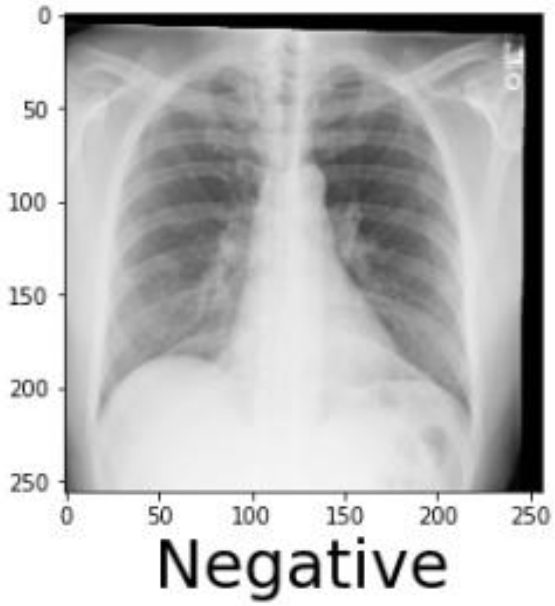
```
<function matplotlib.pyplot.show(close=None, block=None)>
```

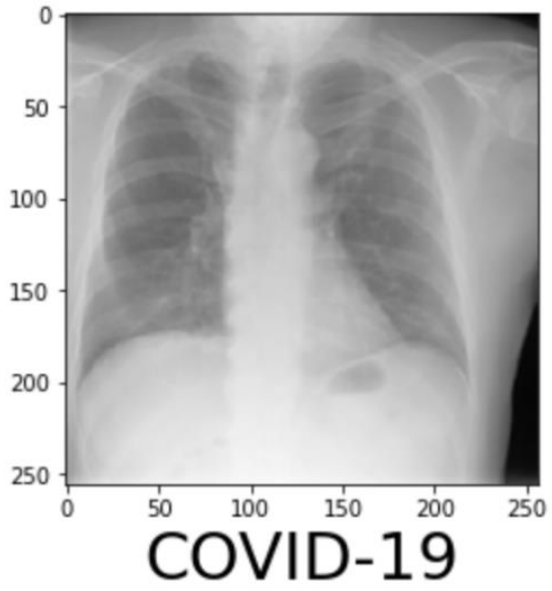


7. RESULTS


0
50
100
150
200
250
0
50
100
150
200
250
COVID-19


0
50
100
150
200
250
0
50
100
150
200
250
Negative


0
50
100
150
200
250
0
50
100
150
200
250
Negative


0
50
100
150
200
250
0
50
100
150
200
250
COVID-19

8. REFERENCES

1. M. Dur-e-Ahmad and M. Imran, “Transmission Dynamics Model of Coronavirus COVID-19 for the Outbreak in Most Affected Countries of the World,” International Journal of Interactive Multimedia and Artificial Intelligence, vol. In Press, no. In Press, pp. 1-4, 2020.

2. S. J. Fong, N. D. G. Li, R. Gonzalez-Crespo and E. Herrera-Viedma, “Finding an Accurate Early Forecasting Model from Small Dataset: A Case of 2019-nCoV Novel Coronavirus Outbreak,” International Journal of Interactive Multimedia and Artificial Intelligence, vol. 6, no. 1, pp. 132- 140, 2020.

3. Q. Li, X. Guan, P. Wu, X. Wang, L. Zhou, Y. Tong, R. Ren, K. S. M. Leung, E.H.Y. Lau, J. Y. Wong and others, “Early transmission dynamics in Wuhan, China, of novel coronavirus–infected pneumonia,” New England Journal of Medicine, 2020.

4. D. Wang, B. Hu, C. Hu, F. Zhu, X. Liu, J. Zhang, B. Wang, H. Xiang, Z. Cheng, Y. Xiong, Y. Zhao, Y. Li, X. Wang and Z. Peng, “Clinical Characteristics of 138 Hospitalized Patients With 2019 Novel Coronavirus–Infected Pneumonia in Wuhan, China,” JAMA, vol. 323, pp. 1061-1069, 3 2020.

5. S. Chauvie, A. De Maggi, I. Baralis, F. Dalmasso, P. Berchialla, R. Priotto, P. Violino, F. Mazza, G. Melloni and M. Grosso, “Artificial intelligence and radiomics enhance the positive predictive value of digital chest tomosynthesis for lung cancer detection within SOS clinical trials,” European Radiology, p. 1–7, 2020.

6. G. Chassagnon, M. Vakalopoulou, N. Paragios and M.-P. Revel, “Artificial intelligence applications for thoracic imaging,” European Journal of Radiology, vol. 123, p. 108774, 2020.

7. F. Song, N. Shi, F. Shan, Z. Zhang, J. Shen, H. Lu, Y. Ling, Y. Jiang and Y. Shi, “Emerging 2019 Novel Coronavirus (2019-nCoV) Pneumonia,” Radiology, vol. 295, pp. 210-217, 2020.

8. J. C. L. Rodrigues, S. S. Hare, A. Edey, A. Devaraj, J. Jacob, A. Johnstone, R. McStay, A. Nair and G. Robinson, “An update on COVID-19 for the radiologist-A British society of Thoracic Imaging statement,” Clinical Radiology, 2020.

9. J. Wu, J. Liu, X. Zhao, C. Liu, W. Wang, D. Wang, W. Xu, C. Zhang, J. Yu, B. Jiang and others, “Clinical characteristics of imported cases of COVID-19 in Jiangsu province a multicenter descriptive study,” Clinical Infectious Diseases, 2020.

10. F. Shi, L. Xia, F. Shan, D. Wu, Y. Wei, H. Yuan, H. Jiang, Y. Gao, H. Sui and D. Shen, Large-Scale Screening of COVID-19 from Community
    a. Acquired Pneumonia using Infection Size-Aware Classification, arXiv preprint arXiv:2003.09860, 2020.

11. S. Wang, J. M. Bo Kang, X. Zeng and M. Xiao, “A deep learning algorithm using CT images to screen for Corona Virus Disease COVID-19),” medRxiv, 2020.

12. L. Wang and A. Wong COVID-Net A Tailored Deep Convolutional Neural Network Design for Detection of COVID-19 Cases from Chest Radiography Images, arXiv preprint arXiv:2003.09871 ,2020.

13. E. E.-D. Hemdan, M. A. Shouman and M. E. Karar, COVIDX-Net A Framework of Deep Learning Classifiers to Diagnose COVID-19 in X-Ray Images, arXiv preprint arXiv: arXiv:2003.11055, 2020.

14. J. P. Cohen, P. Morrison and L. Dao, COVID-19 Image Data Collection, arXiv preprint arXiv: arXiv, arXiv:2003.11597, 2020.

15. RSNA Pneumonia Detection Challenge. Kaggle. [online] Available at: https://www.kaggle.com/c/rsna-pneumonia-detection-challenge, Accessed 29 April 2020.

16. M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman, “The pascal visual object classes (voc) challenge,” International journal of computer vision, vol. 88, p. 303–338, 2010.

17. W.Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu and A. C. Berg, “SSD Single Shot MultiBox Detector,” Lecture Notes in Computer Science, p. 21–37, 2016.

18. K.Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” CoRR, vol. abs1409.1556, 2014.

19. H. Jansen, Radiología dental. Principios y técnicas., Mc Graw Hill, 2002.

20. A. M. Reza, “Realization of the contrast limited adaptive histogram equalization (CLAHE) for real-time image enhancement,” Journal of VLSI signal processing systems for signal, image and video technology, vol. 38, p. 35–44, 2004.

21. H.-W. Ng, V. D. Nguyen, V. Vonikakis and S. Winkler, “Deep learning for emotion recognition on small datasets using transfer learning,” in Proceedings of the 2015 ACM on international conference on multimodal interaction, 2015.

22. Q. Sun, Y. Liu, T.-S. Chua and B. Schiele, “Meta-transfer learning for few-shot learning,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2019.