(LUNG AND COLON CANCER PREDICTION USING CNN)

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

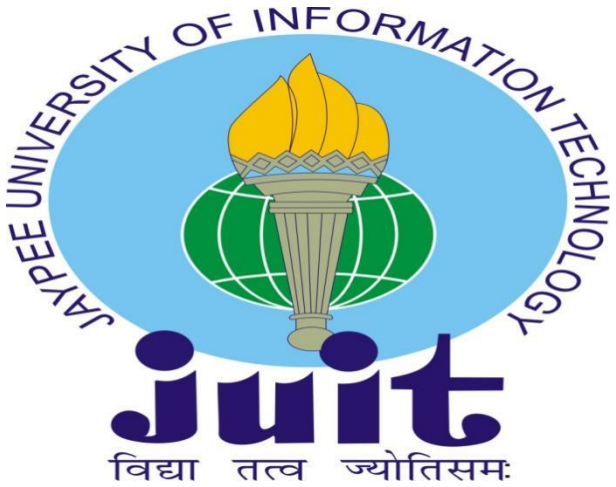
By

(Harsh Singhal (181318))

Under the supervision of

(Dr. Monika Bharti)

to



Department of Computer Science & Engineering and Information Technology
Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"Lung Colon Cancer Prediction using CNN"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2022 to May 2022 under the supervision of **(Dr. Monika Bharti)** (Assistant Professor and Computer Science).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Harsh Singhal,181318

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Dr. Monika Bharti
Assistant Professor
Computer Science
12/05/2022

Acknowledgement

Firstly, I express my heartiest thanks and gratefulness to Almighty God for his divine blessing that makes it possible to complete the project work successfully.

I am grateful and wish my profound indebtedness to supervisor Dr. Monika Bharti, Assistant Professor S.G., Department of CSE Jaypee University of Information Technology, Waknaghat. Deep Knowledge & keen interest of my supervisor in the field of "Deep Learning, Machine Learning, and Neural Networking" helped me to carry out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts, and correcting them at all stages have made it possible to complete this project.

I would like to express my heartiest gratitude to Dr. Monika Bharti**,** Department of CSE, for his kind help to finish my project.

I would also generously welcome each of those individuals who have helped me straightforwardly or in a roundabout way to make this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Harsh Singhal
181318

Table of Content

List Of Abbrevations

CNN : Convolutional neural network

ILSVRC : Large Scale Visual Recognition Competition

NSCLC : Non-small cell cancer of the lungs

SCLC : Small cellular cancer of the lungs

GLCM : Gray Level Co-occurrence Matrix

VGG : Visual Geometry Group

CAD : Computer Aided Detection

SGD : Stochastic Gradient Descent

List of Figures

List of Tables

Abstract

Colon and Lung cancer is one of the most perilous and dangerous ailments that individuals are enduring worldwide and has become a general medical problem. To lessen the risk of death, a legitimate and early finding is particularly required. In any case, it is a truly troublesome task that depends on the experience of histopathologists. If a histologist is under-prepared it may even hazard the life of a patient. As of late, deep learning has picked up energy, and it is being valued in the analysis of Medical Imaging. This paper intends to utilize and alter the current pre-trained CNN-based model to identify lung and colon cancer utilizing histopathological images with better augmentation techniques. In this paper, eight distinctive Pre-trained CNN models, VGG16, NASNetMobile, Xception, VGG19 are trained on LC25000 dataset. The model performances are assessed on precision, recall, f1score, accuracy, and auroc score. The results exhibit that all eight models accomplished noteworthy results ranging from 96% to 100% accuracy.

Chapter 1: Introduction

1.1 Introduction

Cancer is a disease that leads when some abnormal cells increases uncontrollably in any organ(i.e lung,colon etc) of the body.Cancer is the 2nd most prominent reason for the death of the population globally which is resulting in around 9.6 million of death in 2018. Colorectal cancer was resulting in around 1.80 million cases of cancer and seven lakh and eighty three thousand deaths, whereas lung cancer results in around 2.06 millions of disease cases & 1.76 millions of people lost their life.

Two kinds of lung cancer that grow and spread quickly are “Small cellular cancer of the lungs (SCLC)” and “non-small cell cancer of the lungs (NSCLC) ”. SCLC is a kind of cancer that highly leads to death and is made up of cells with neuroendocrine properties that accounts for 15percent of all lung cancer cases. NSCLC is further divided into three major pathologic subtypes: adenocarcinoma , squamous cell carcinoma , and enormous cell carcinoma , which account for the remaining 85percent of cases. Colon cancer is commonly referred to as colorectal cancer ,which is made up with two words both colon and rectal cancers that begin in the rectum. Adenocarcinoma accounts for 96percent of all instances of colorectal cancer.The major focus of this research is on non-small cell lung tumours (NSCLC) .

Researchers are developing histopathologists’ labour, that involves looking at cells and tissues under a microscope and identifying irregularities. In recent years, a great deal of effort has gone into scanning the entire tissue and saving the work as a digital image. All the work results in a massive amount of WSI (full slide pictures) is being gathered. Attempts have been made to investigate WSIs using machine learning techniques for diagnosis [10]. Since the victory of the group using deep learning in the Imagenet “Large Scale Visual Recognition Competition” (ILSVRC) in 2012, the huge amount of pictures gathering assignments have been shifted to deep learning. All the above work which have been done could be summarized in few points which are as follows:

1) The data present in the dataset are divided into different classification.

2) Information augmentation by the use of “imgaug” library.

3) Feature Extraction through transfer learning using various already trained CNN models

4) Metrices were used in order to get the evaluation of the results.

5) Conclusion and future work.

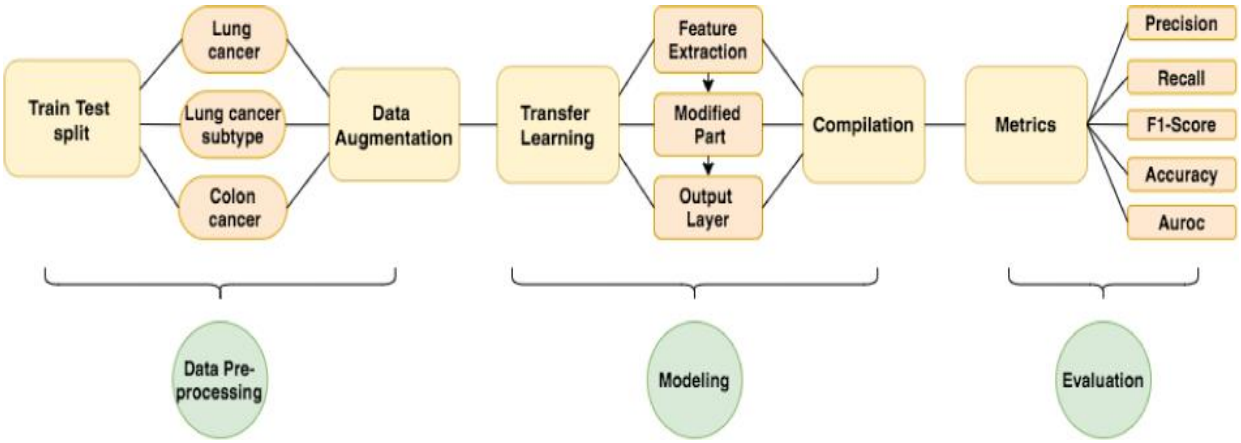


Figure 1: Flow Chart of Prediction of Lung and Colon cancer

1.2 Problem Statement

Most cancers (for example, colon and lung cancer) are among the most risky and severe illnesses that people face worldwide, and they have become a major scientific concern. A valid and early location is especially necessary to lower the risk of death. In any event, it's a very difficult task that relies on the expertise of histopathologists. If a histologist is unprepared, a patient's life may be jeopardised. Deep researching has regained popularity recently, and it's now being appreciated in Medical Imaging research.This work aims to utilise and improve the present Convolution neural model for detecting lung & colon tumours utilising histopathology images and greater augmentation techniques. On the LC25000 dataset, 8 unique Pretrained CNN models, vgg16 , NASNetMobile , and vgg19 , are trained. Precision, keep in mind, f1score, accuracy, and auroc rating are used to evaluate the version's efficiency. The results demonstrate that each of the four models had notable results, ranging from 96percent to perfect certainty(100%). GradCAM & SmoothGrad are then utilised to visualise the attention pictures of Pre-educated CNN models that identify malignant & benign pictures.

1.3 Objectives

Colon and lung cancer is the most serious & deadly diseases that people face across the world, and it has become a widespread medical issue. A valid and early discovery is especially necessary to reduce the risk of mortality. In any event, it is a difficult process that relies heavily on the expertise of histopathologists. If a histologist is unprepared, a patient's life may be jeopardised. Deep learning has recently gained popularity, and it is now being appreciated in medical imaging analysis.

The goal of this study is to improve on the present already trained CNN models for the detection of lung & colon cancer using histopathology images and improving the data images using Augumentation technique which results in the increase in the data size. The LC25000 dataset is used to train four different pre-trained CNN models: VGG16 , NASNetMobile , Xception and VGG19 .

Precision , recall , f1score, accuracy , and auroc values are being used to receive the cnn model's performance. The results show that all eight models achieved notable accuracy levels that varies from 96percent to 100percent. GradCAM and SmoothGrad are then utilised in order to visualise the pictures of predefined CNN models that have been trained to distinguish malignant and benign images.

1.4 Methodology

Figure 1 illustrates and defines the categorization structure used in this study. The framework is broken down into three sections: data preprocessing, modelling, and evaluation.

1.4.1 Train Test Split

The train test split procedure is often used to assess the working of the ML algorithms which provide results on information that was not used to train a model.
It is a fat and easy method for evaluating the efficiency of various ML algorithms for predictive modelling problems. Although the procedure is very simple to use & understand, there really are times when it should not be used, such as when we only use a small dataset or when further preparation is required, such as when it is used for categorization and the database is unbalanced.

The dataset used in the study of prediction is LC25000 which consists histopathological images of lung & colon.The dataset was released to address the lack of availabilty of data regarding lung and colon in medical imaging.It includes Fifteen thousand lung and colon images which are subdivided into 5 categories: lung adenocarcinoma , lung squamous cell cancer, lung benignant , colon carcinoma , & colon benign . Each class has 5000 photos and data are divided in the ratio of 80 and 20 for 3 categorization problems:

1) A sum of 5000 images of lung benignant & adenocarcinomas and squamous cell carcinomas were used in a binary classification to diagnose lung cancer.
2) Lung cancer subtypes are categorized to both adenocarcinoma and squamous cell carcinoma using 10000 pictures.
3) One more binary categorization for locating colon most cancers from a hard and fast of 5000 colon carcinoma and 5000 non-cancerous colon pix.

The splitting of pictures for categorization utilised in this study is shown in Table 1. acc/cc = adenocarcinoma /carcinoma , scc = squamous cell carcinoma , ben = benignant are the column titles in Table 1.

Table 1. Overview of the distribution of LC25000 image dataset

Classification task	Training set 80%			Test set 20%		
	acc\cc	scc	ben	acc\cc	scc	ben
Lung cancer	2000	2000	4000	500	500	1000
Lung cancer subtypes	4000	4000	–	1000	1000	–
Colon cancer	4000	–	4000	1000	–	1000

1.4.2 Data Augmentation

Data augmentation is a collection of techniques for generating extra data from current data in order to increase the amount of data available. This could include making minor/small changes to the information which is already available or by using deep learning models to generate more data. Data augmentation may assist improve the performance and output of machine learning models by producing new and varied cases to train datasets.

Whenever the information is large and diverse, a ml model performed best and much more accurately. Information augmentation is used in data analysis to describe methods for increasing the amount of information present by bringing slightly changed duplicates of known information or developing better synthetic information from the given information. It acts as a regularizer while training a ML model, reducing overfitting . It has a very close link to interpolation in data analysis.

The photos obtained with LC25000 are 768*768 pixels in size. From the original 768 × 768 pixels, all photos have been minimized to a box (square) proportions of 224 x 224 pixels. The following augmentations were already applied to the pictures in the LC25000 image dataset:

1) Left direction rotation and right direction rotations.
2) Horizontal and vertical flips.

However, to improve the results of the models, this study employs a complicated augmentation pipeline from the imgaug library. Figure 2 shows an original as well as a collection of 64 enhanced photos of colon &lung cancer from the LC25000 pictures database, created with the imgaug package (library) .

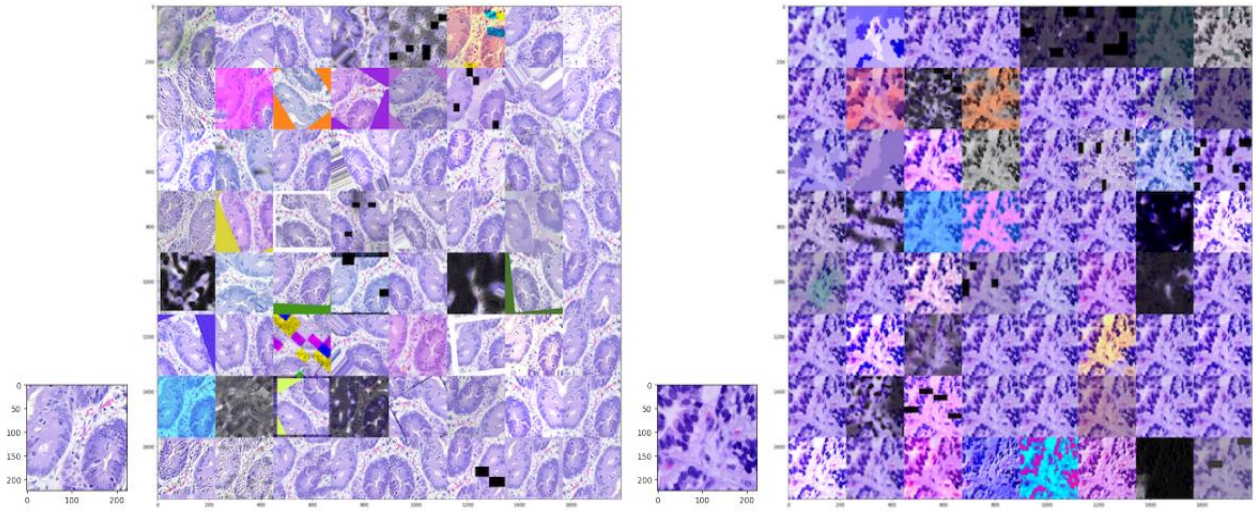


Fig. 2. Original and 64 augmented images using imgaug library of both colon and lung carcinoma respectively.

1.4.3 Transfer Learning

There are certain pretrained models available which saves our lot of time to train data for evey new problem and usage of these data models is known as Tranfer Learning.It's especially popular at the moment now in deep learning because it can learn deep neural networks with very little input. This is beneficial in data science because most real world problems does not require lot of labelled data to train complicated models.

With growing number of online users, there is an abundance of data to analyse. Dealing with such massive volumes of data in the next years will be challenging [31] because there are some security and privacy issue with database so datset are scarce. Histological images, are also very variable. It would also be difficult and time taking to build a model starting scratch with randomly initialised characteristics and limited data observations. To solve these issues, the researchers used pre-trained Cnn architectures. The purpose is to solve additional problems with the weights from one model. This is also known as learning algorithms. It aids in the decrease of computational work.

The structure contains three sections:
1) Feature Extraction Part -Training of well-known Pre-trained CNN architectures to extract the most significant characteristics and fine-tune the models. Following this, these traits would be moved to the changed portion for more categorization.

2) Modified Part -To achieve a vector of features, adding a concatenate layer, which is a combination of three layers that consolidates Max-pooling2D, Average-pooling2D, and a flatten layer. In addition, a dropout layer with a 0.5 dropout rate is also included.

3) Output Section -A sigmoid activation is used for adding some more features in the output layer of the model and the function is expressed as:

$$f(x) = \frac{1}{1 + e^{-x}} \quad \text{------(1)}$$

1.4.4 Compilation

The compilation of the model requires an optimization algorithm. There are various optimization algorithms such as adaptive learning rate (Adam), stochastic gradient descent (sgd), Adadelta, RMSProp, AdaMax, and many more. We picked Adam as our optimization as it is the standard most and works commendably well for any kind of deep learning model [41]. The hyperparameters values used for training the models are: Learning rate = 0.0001, epochs = 10, Batch size = 64.

1.4.5 Metrics

The fashions can be dependent on precision, take into account, f1-score , accuracy , and auroc score. All these features help in picking up the best suited model. So,Let the letterr be:

(Tp) = Number of times model predicts the positive class correctly.

(Tn) = Number of times model predicts the negative class correctly.

(Fp) = Number of times model predicts the positive class incorrectly.

(Fn) = Number of times model predicts the negative class incorrectly.

$$Precision = \frac{T_p}{T_p + F_p} \quad (2)$$

$$Recall = \frac{T_p}{T_p + F_n} \quad (3)$$

$$Accuracy = \frac{T_p + T_n}{T_p + T_n + F_p + F_n} \quad (4)$$

$$F1 - Score = 2.\frac{Precision.Recall}{Precision + Recall} \quad (5)$$

Precision , recall , accuracy , and f1score can be calculated in this way.
The region underneath an area operational curves (Auroc) depicts the relationship among true positive and false positive rates at approaches suggest.

1.5 Organization

The project report is broken down into 5 sections. The first chapter covers the background and motivation for the proposed application, the problem statement and aims to answer the issue statement, the recommended technique or research, and the highlighting of successful proposed applications. Chapter 2 illustrates the literature survey of the project from which we took the references. The system development chapter includes the site map, use case diagram, activity diagram, and system wireframe, which is the proposed application's user interface. Software design approach, tools, requirements, system performance specifications, and timescales are discussed in Chapter 4. The fifth chapter concludes the implementation, project evaluation, benefits and future scope of the project.

Chapter 2 : Literature Survey

A fine quantity of work has been done for the categorization of histopathological images of variety of cancer types such as breast, lung, colorectal, skin cancer etc.

The Gray Level Cooccurrence Matrix (GLCM) technique was used by Liping Jiao et al. [11] to extract eighteen regular features, including grayscale suggest, grayscale variance, and sixteen texture functions. On 60 colon tissue snap photos partitioned similarly into the two lessons, an SVM based classifier was applied, which concluded precision, F1-rating, and account for ninety six.67%, eighty three.33 percent, and 89.Fifty one percent, respectively.

Scott Doyle et al. [13] created thirty four hundred functions using 48 breast tissue sections using textural and architectural-based totally skills (thirty malignant, eighteen benign). Furthermore, spectral clustering has been used to reduce the size of the feature set. After that, an SVM classifier has been used to binary categorise malignant & non cancerous breast tissue images and discriminate between images with a low and high degree of malignancy. These results indicate that using textural features, 95.8 percentage efficiency was achieved in distinguishing cancerous tissues from non-malignant development. The reliability of 93.3% in detecting the high degree from the low level of most malignancies has been achieved by incorporating characterstics.

S. Rathore et al. [14] presented a feature extraction strategy that mathematically shape the structural properties of the elements of colon tissues. To expand a mixed characteristics collection, conventional factors are provided with morphological, texture, scale-invariant feature remodel (SIFT), and elliptic Fourier descriptors (EFDs). SVM [12] is then applied as a classifier on 174 colon biopsy pictures, achieving in a 98 percentage overall accuracy.

The most serious problem which academics around the world face is a lack of information in the scientific field, which leads to overfitting [15] of trends. To avoid overfitting, F. A. Spanhol et al. [16] used a variety of data augmentation approaches to expand the amount of input pictures on the BreakHis [18] database. The AlexNet [6] variant is then trained from very begining which resulting in an efficiency of ninety percent.

Hiba Chougrad et al. [19] explore the effects of transfer learning [6] by determining a technique(fine-tuning) when training an InceptionV3 [20] model. Combining several databases such as the DDSM database [21], the INbreast database [22], and the BCDR database [23] yields a bigger database with an efficiency of 98.845 percent.

Sara Hossein Zadeh Kassani et al. used a total of four data information sources, including BreakHis [18], ICIAR [32], PCam [33], and Bioimaging [34], to outfit well-set up Pretrained CNN architectures, i.e VGG19, MobileNetV2, and DenseNet201 [30] for aided diagnosis of breast most cancers identification. With the exception of the Bioimagingdataset, which achieves just 83.1 percent, all databases achieve better than 95%.

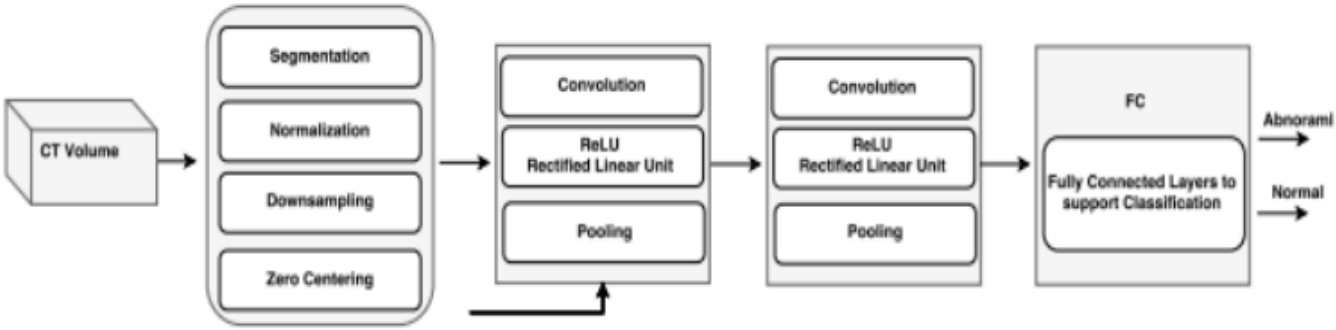


Figure 3: CNN Architecture by Sara Hossein Zadeh Kassani et al.

Wafaa Alakwaaet al. suggested a three-dimensional Convolution neural base technique to find lung tumours in 2017. The databases Kaggle Data Science Bowl and LUNA16 were utilised. Because lung nodules also weren't classified in the Kaggle database, LUNA16 was used to educate the U-Net version to detect them. The picture pre-processing step included segmentation, downsampling, normalisation, and zero centered. The CT pictures' pixel values were first converted into Hounsfield devices, and then thresholding was utilised for classification. Following segmentation, 3-D picture normalisation was completed with the goal of mapping values between 0 and 1. Zero downsampling. In all 3 components, 5 items have been produced.Lastly, the average score of the pictures were subtracted from the training dataset to accomplish zero centering. To determine the actual location of nodules, a U-Net was trained using the LUNA16 dataset instead of simply putting the split pictures further into classifier. The accuracy, false-positive rate, Misclassification rate, and untrue negative rate was determined to be 86.6 percent, 11.9 percent, 13.4 percent, & 14.7 percent, respectively. Figure depicts the 3D CNN's fundamental design.

Using two hundred and ninety eight lungs pictures, Tetsuya Tsukamoto et al. [35] created a deep cnn model of three convolutional three pooling layers and two fully connected layers for multiple classification tasks. Various data augmentation techniques such as rotations, zooming, and flipping was applied to avoid fitting problem [15]. The accuracy of adenocarcinoma, squamous cell carcinoma, and little cell carcinoma is 89.0 percent, 60.0 percent, and 70.3 percent, accordingly, in the findings obtained by controlling of enhanced photos, for a complete certainty of 71.1 percent.

During the year 2019, Samaiya Dabeer et al. suggested utilising a Convolution neural technique to diagnose cancer in a histological picture. BreakHis, MITOS- ATYPIA14, and the Original Data Set (UC Irvine ML Collection). It was determined that the BreakHis datasets network will be used. The model has been trained using the RGB colour model, which contained 2480 benign & 5429 malignant data. As a consequence, the recommended method employs an effective categorization model to distinguish between benign and malignant breast tissue. The deep net was then constructed by analyzing the photographs in the database. Information Duplication should be deleted because it causes network

problems and therefore is obsolete. The efficiency for the benign outcomes judged to be 90.55 percent and 94.66%, accordingly.

During 2019 Pouria Moradi et al. [27]. specifies using 3D CNN as to order to reduce the false possitives. Xavier weight initialization is used to initiate the networking parameters. Stochastic gradient descent is used to train networking parameters with a knowledge price of 0.01 and approximatelt 10 to 5 decay for each epoch having zero point nine momenta. A Meta classifier was created by combining 3 decision tree that had been done with training. The LUNA 16 database was used in training & assessing the system. For 3.09 false positives, the system has a 91.23 percent accuracy rate.

Moradi et al. in 2019 examined several approaches for distinguishing lung cancer nodules from non-nodules.They developed the 3D CNN technique to decrease or remove non positive precessions since nodules come in different size, utilising simply 1 CNN might lead to erroneous detections. In the results,the nodules were categorized to 4 groups depending on their size. They also employed four different 3D CNN sizes. To improve the results, they integrated all four classifiers. Each CNN is made up of a number of 3D CNNs of varied sizes. In order to get superior results, all 4 classifiers were integrated. Each CNN was created using mixing of Max pooling layers and convolutional layers. ReLU is the activation function employed here. Finally, a Softmax layer with a completely linked layer is employed to generate the output. Because nodules range in size from 3mm to 3cm, a single layer estimate might be incorrect for either very tiny nodules or very big values. So they combined the output values (predicted values) of all four CNNs and submitted them to a final classifier. They picked a logistic regression classifier that accepts four CNN inputs and generates a final prediction.So researchers used a decision tree classifier & gradient boosting to execute logistic regression.

Atsushi Teramoto et al. [7] in 2017 developed a Deep CNN-based technique for automatically classifying different type of lung cancer from Medicalised Pictures. Seventysix cancer cell samples are included in the imagenet database. Picture enhancement is performed on pictures generated by microscopy that do have varying & direction invariant clarity of the aimed sites. Filter is applied with a Gaussian filter as well as a neural border augmentation filtered. Every one of the specifics, such as filter size and layer stride, were given. 2 fully connected layers, a convolutional layer, and three pooling layers comprise the structure. With DCNN, 70percent of total of the categorization is accurately calculated.

During 2018, Margarita Kirienko et al. adviced a CNN based strategy with 69percent, 69percent, and 87percent reliability rejection, trial, & test sets accordingly. Lung cancer was staged from 1 to 4 using Tumor, Node, Metastasis (TNM) staging. The pictures were obtained using fluorodeox glucose positron emission tomography (FDG-PET) and CT. CNN's usage was to classify those pictures into "T1-T2" or "T3-T4" categories. This system was designed using multiple networks: a classifier and a feature extractor. The feature extractor was used to extract relevant features, and the patch was classified using a classifier. Four Hundred and seventy two participants were involved in the study.

Qinghai Zhang et al. [30] suggested a automated technique towards constructing a lung nodule alert system in 2020. The LIDC-IRDI public database was utilised to test the suggested approach. This study's proposed technique is the MultipleScene DL Environment, that consists of multiple phases. The probability distribution of discrete grey

levels is determined via threshold segmentation, which is Histogram, using CT scans as input. The major goal of the lung parenchyma segmentation method is to correct the smooth lung outlines. The exchane of the pulmonary vein system aids in the identification of the nodule structure. The vessels are removed using vessel filters, which reduces the amount of false positives.CNN's design includes a poling layer, a convolutional layer, as well as a completely integrated layer. There are two categories of picture data: Class 1 and Class 2 and discrete pictures that are isolated from lung images using parts and classification [31]. To detect malignant tumour cells in the lungs, segmentation is used.

According to the research, only just few studies have been done to categorise specific malignancy types which is to categorized to lung cancer and its subtypes as well as colon & breast cancer too. The much more Pretrained Cnn models there are, the fewer models there would be to compare.

Bohdan Chapliuk et al. used C3D & 3-dimensional DenseNet neural network models to identify lung cancer using CT scans in 2018. These neural networks were tested using whole lung three - dimensional images and 2-stage approaches which were used for breaking to parts and categorization, 2 different neural networks are examined.) and then examined once again. The Data Science Bowl 2017 database was used, which included CT scans of over 1000 patients. All of the CT images were resampled into Household Units (a measure that describes xray intensity) for pre-processing. Because HU ranges are unique to tumours (-500), all patient photos were filtered out in the second stage by a range for lung tissue that filters out all bones from the image. The 3D patient picture was shrunk to 120x120x120 pixels. All 3CD & 3-dimensional DenseNet offer comparable findings, with 3-dimensional DenseNet surpassing 3CD by a little margin. In comparison to two methods, Neural Networks examines on complete lung 3-dimensional pictures conducted worse.

Mesut Togacar et al. [32] presented a Convolution neural based lung cancer prediction system in 2020.They received a sum of hundred pictures from 69 different patients (50 malignant and 50 non-cancerous). Because there were fewer photos, agmentation was utilised to produce a good dataset. To change the parameters for every examining data, Stochastic Gradient Descent was utilised as an optimization approach. In addition, the optimization algorithms RMSProp and ADAM were applied (for LeNet). The characteristics were extracted using the mRMR technique.Following the CNN designs, classic ML models such as LR, LDA , SvM, KnN, and DT are utilised. The Principal Component Analysis approach was used to improve the performance. Using KNN with CNN and mRMR, 99.51percent accuracy was received.

Utilizing deep CNN-based approaches, Mehdi Fatan Serj et al. [26] suggested a methadology to identify lung tumours effectively in 2018. 2 max-pooling levels, 3 convolutional layers, a softmax (binary) layer, as well as a completely connected layer comprised the network. For the Kaggle Information Research Bowl 2017 contest, the technique was tested using Kaggle's database. Other CNN-based models scored worse than the deep Convolution neural model.To optimise the multinomial logistic regression goal and hence the likelihood of patients with lung cancer, cross-entropy was utilised for the loss function. They attained a sensitivity of 87 percent and a specificity of 99.1 percent.

A major pressing issue confronting academics throughout the globe is a lack of information in the healthcare setting, which leads to model overfitting . To prevent fitting

problem, F. A. Spanhol et al. used several information augmentation procedures to raise the amount of input photos in the BreakHis database. The AlexNet model then was taught from start, resulting in an accuracy of 90percent.

Chapter 3 : System Development

3.1 CNN

Convolutional neural networks are specialised neural networks which are meant to collect localised input. By directly integrating image (2D) information, and also by extension, one- dimensional or 3-dimensional information, in the architecture, CNNs is designed to effectively manage such information. Convolutional neural networks are amongst the neural network types that has aided the advancement of machine learning . CNNs are a type of deep neural network which is most basically used in analysis of visual data.

The convolutional layer, which can be incorporated in a deep neural network, has propelled deep learning to make substantial advances in image analysis. Whenever a network contains a convolutional layer, it is recognised as a CNN. A CNN's major characteristic is its ability to recognise picture properties including light, darker, or particular colour patches, borders in multiple orientation, clusters, and much more..

A CNN could identify other complex traits such as with a mouse's ears, a puppy's snout, an user's eyes, or particular forms by employing these more fundamental aspects. This goal of recognising additional characteristics based on the pixels of the input image is exceedingly challenging for a standard neural network. Inside this picture, the characteristics might appear in various places, rotations, and proportions. Even though the thing seems to be the same with our eyesight, any movement of the object or camera angle might lead the number of pixels to substantially shift.

Since every pixels in a picture represents a possible "feature," utilising a fully linked network would result in overfitting because to the large number of parameters to consider. The presumption of a hierarchy structure in the dataset is made using a convolutional neural network.

To provide a (excessively) basic example, a group of pixels might create borders, a collection of edges could create a certain part of the body, and a collection of internal organs could resemble a specific creature Because CNNs account for these spatial links, the set of parameters in the design is decreased, lowering the risk of overfitting.

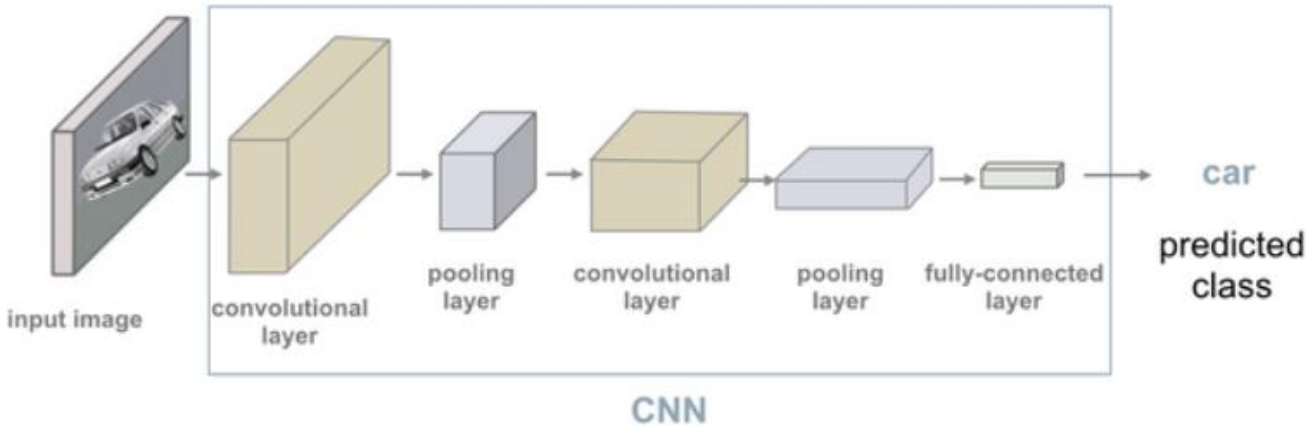


Figure 4: CNN Model Description

A convolutional neural network architecture generally has several components:

1) **Convolution layer** -We had numerous convolutional layers in a network that extract low to high-level features depending on which layer that were concentrating on. Early convolutional layers pick up lower-level features (such as bars & corners), whereas subsequent convolutional layers pick up even higher characteristics based on the parameters from relatively low characteristics (like forms and structures) - similar to how sight functions inside the human mind.

2) **Pooling layer** - For picture categorization, convolutional neural networks are commonly utilised. Unfortunately, because photos were high-dimensional information, we'd rather limit its dimensions to avoid the risk of overfitting. Pooling decreases the spatial dimensions of a picture by using numerical methods like average or max-pooling A Pooling has been commonly used because it works as a noise suppressor, makes picture categorization invariant to functioning, and allows us to collect fundamental structural aspects of the represented images without being bogged down by minor things.

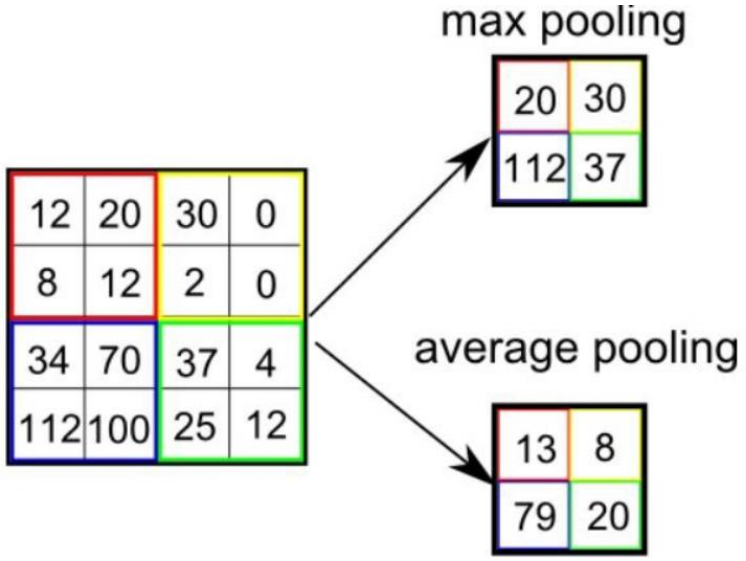


Figure 5: Max and Average Pooling

3) Fully-connected layer - Before to transmitting this information over to the fully connected (Dense) layer, a sequence of convolutional and pooling layers procedures may be thought of as dimensional reduction stages. When undertaking categorization, the fully connected layer takes the "compressed" representation of the picture and tries to fit a simple NN (multi-layer perceptron) into it.

From around 1980s, Cnn models had first been created and then used. At the time, the best

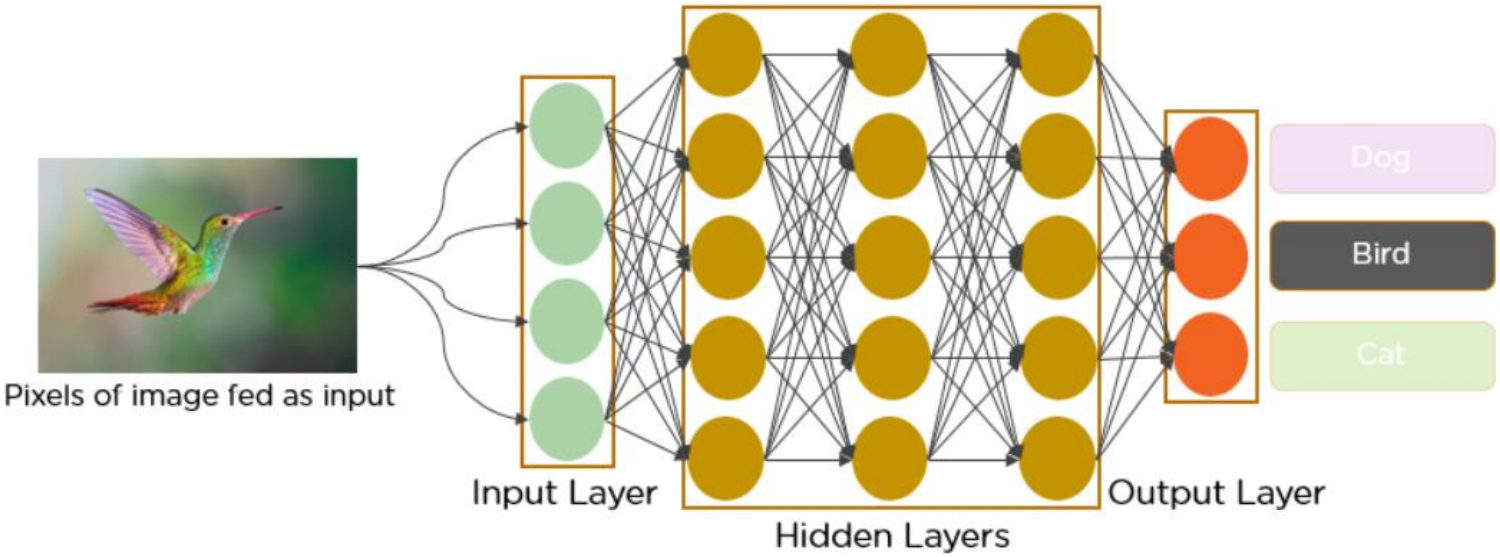


Figure 5: Layers in CNN

a CNN could do is to detect handwritten characters/digits. It was largely used by the postal service to detect postcodes, pin codes, as well as other related data. The most key point to remember about any deep network is that it requires a lot of information & computational capacity to construct and above mentioned problem become a disadvantage for the cnn since it become restricted to the field of machine learning.

3.2 Imagenet

The ImageNet is a visual dataset designed to help in the growth of visual object tracking applications. Approximately fourteen million photos have been hand annotated to identify what is displayed, with bounding boxes supplied in at least one million of them. ImageNet has around 20,000 categories, with a simple category like "balloon" comprising several hundred images. ImageNet gives a free library of annotations for third-party image URLs, while the images itself are not controlled by ImageNet.The ImageNet have been hosting a annual software contest, “The ImageNet Large Scale Visual Recognition Challenge”, since 2010, In this comptetion various softeware applications compete that how much accurately it could recognize scenes & objects.

ImageNet is a database that consists of different hundreds of thousands of images that are arranged in order of wordnet hierarchy. Imagenet have alse played a crucient role in deep learning and computer vision and data in the Imagenet is freely available for researchers.
In Between 2010 and 2017, the ImageNet Large Scale Visual Recognition Challenge, or ILSVRC for short, was an annual competition in which challenge problems used portions of the ImageNet dataset.
The challenge's objective is to encourage the growth of new computer vision techniques while also evaluating the present status.
The yearly comptetion works on a variety of tasks for "image classification," that includes providing a class label to a picture based on the principal item in the shot as well as "object detection," which entails locating things inside the image.
The ImageNet data collection contains 14,197,122 tagged photos, as explained by WordNet hierarchy . The ImageNet “Large Scale Visual Recognition Challenge” (ILSVRC), an image analysis and object detection standard, has been using the dataset since 2010. A collection of manually annotated training photos is included in the publicly available dataset. A series of sample photos is also available, although without any hand comments.
There are two types of ILSVRC:
(1) Picture analysis of a binary label for the appearance of a object class in the picture.
(2) A rigorous frame & class label surrounding an item instance in the picture are annotated at the entity level, Because the ImageNet initiative does not hold the rights to the photographs, just thumbnails and Links are available.

3.3 Keras

Keras uses a pythone interface and is a software library which is useful in artificial neural network and is also freely available to all the users. The TensorFlow library is accessible using Keras . Keras supports a variety of backends including TensorFlow , Microsoft Cognitive Toolkit , Theano , and PlaidML . As of release 2.4, just TensorFlow is enabled. It is designed to be user-friendly, flexible, and scalable, with the goal of allowing fast deep neural network testing. This was developed as part of the "ONEIROS research project", and its main creator and operator is "François Chollet" who is a Google Engineer. XCeption deep neural network model.was invented by Francois.

Keras is having several implementation of standard cnn basic components such as layers, goals, activation functions , optimization techniques, and some other tools are also available for dealing with picture & textual information easier.It also helps in minimizing the amount of code to be written for a neural network.The sources of the code are available on various sites and suppot forums which are github,Slack channel etc. In addition to standard neural networks, Keras offers convolutional neural networks. Dropouts, batch normalisation , and pooling are some other prominent utility layers that are provided. Keras enables deep learning model productization on phones the web, and the Java Virtual Machine.

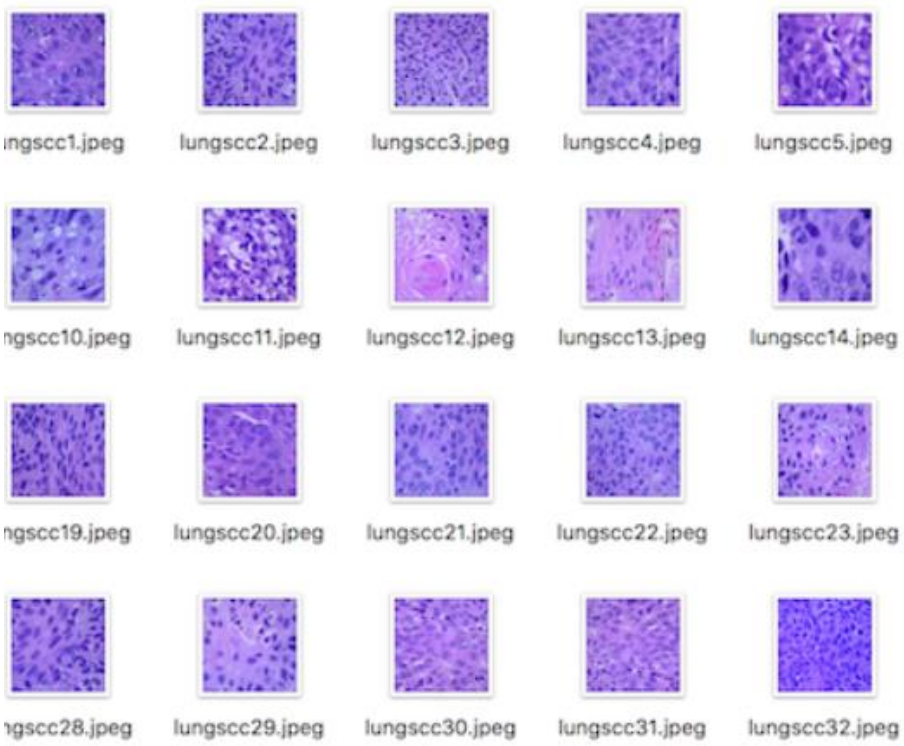
3.4 LC25000 Lung and Colon histopathological Image Dataset

Machine Learning is a type of artificial intelligance which have helped in getting advanced in some variety of fields such as medicine.Machine Learning models requires a huge amount of data for the training purpose. Since database of medical images such as cancer are a larhe requirement nowadays.The given datset ihave twnty five thousand images which are further divided into 5 categorization which are having 5000 each.Data set is divided into two folders namely, colon imagesets and lung one also.Size of the dataset is1.85GB and resolution is 768*768.

The colon image sets subdirectory has two secondary subfolders: colon aca, which has 5000 photos of colon adenocarcinomas, and colon n, which contains 5000 images of benign colonic tissues.The lung picture sets subdirectory has 3 main subdirectories: lung aca, which contains 5000 photographs of lung adenocarcinomas, lung scc, which contains 5000 pictures of lung squamous cell carcinomas, & lung n, which includes 5000 pictures of benign lung tissues.

```
File counts
./lung_image_sets/lung_aca :     5000
./lung_image_sets/lung_n :     5000
./lung_image_sets/lung_scc :     5000
./colon_image_sets/colon_n :     5000
./colon_image_sets/colon_aca :     5000
```

Figure 6: Classification of Dataset



3.5 VGG16

Karen Simonyan and Andrew Zisserman of “Oxford University's Visual Geometry Group” (VGG) introduced VGG models which is a type of Cnn structure which results very good in a comptetion called Imagenet.Researchers put 6 various models to the test, each with its own varying amount of learnable layers. Determined by the number of models available, VGG16 and VGG19 are the two most popular variations.

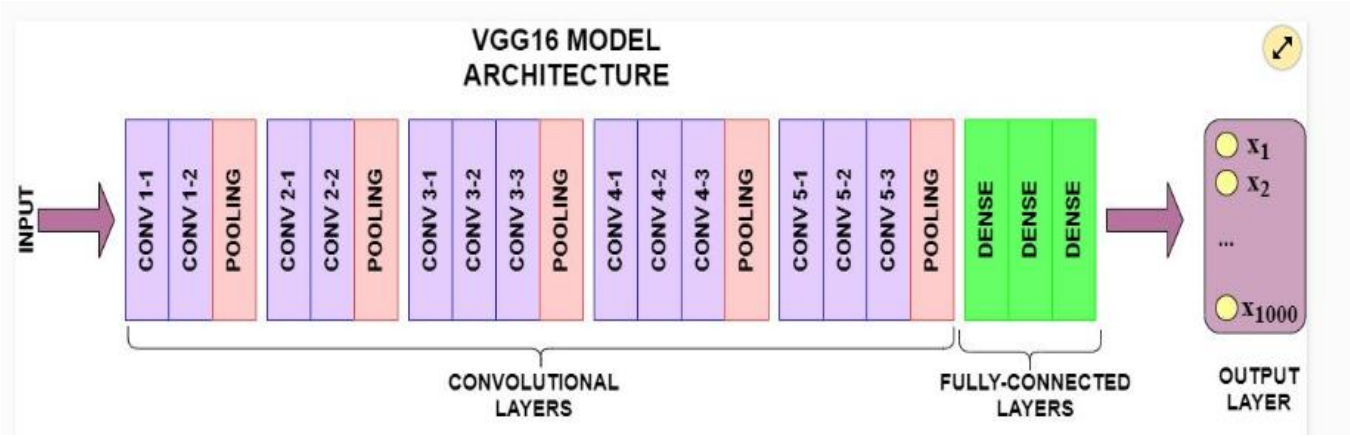
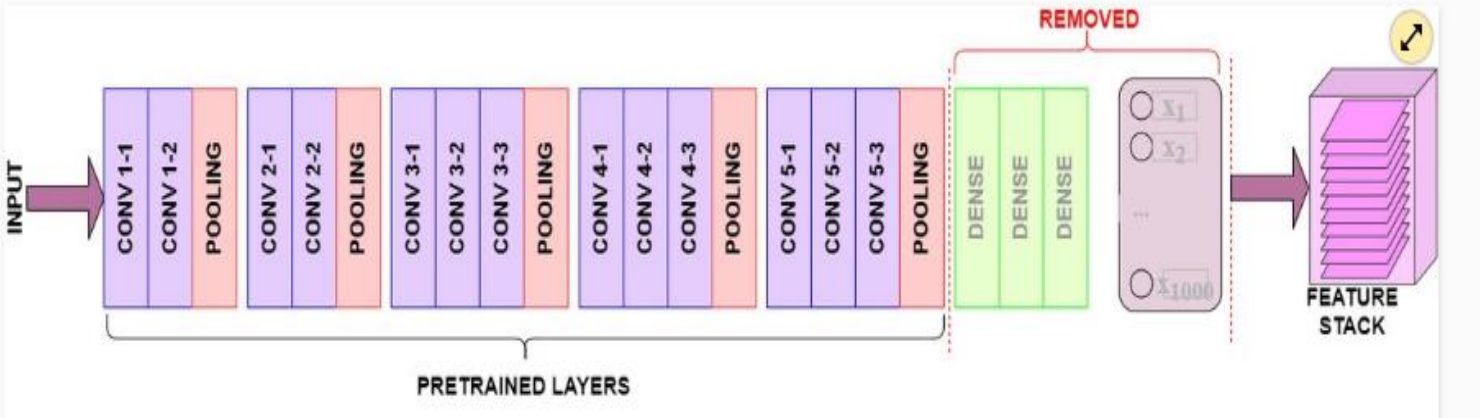


Figure 7: VGG16 Architecture

The ImageNet collection was found to carry photos of various automobiles such as cars trucks etc. One may bring in a cnn model which has been already examined on the Imagenet datasets & retrieve features using already examined layers. We could no longer utilise the complete structure of the already trained model. Their Targeted Data has only two classes for predictions, however the Completely connected layer provides 1,000 different output labels. So, We are going to take a already trained models such as Vggsixteen and "chop off" a completely Connected layer (also known as "top" model). We could use one of two Transfer Learning algorithms that once pre-trained layers have been imported, except the "top" of a model.

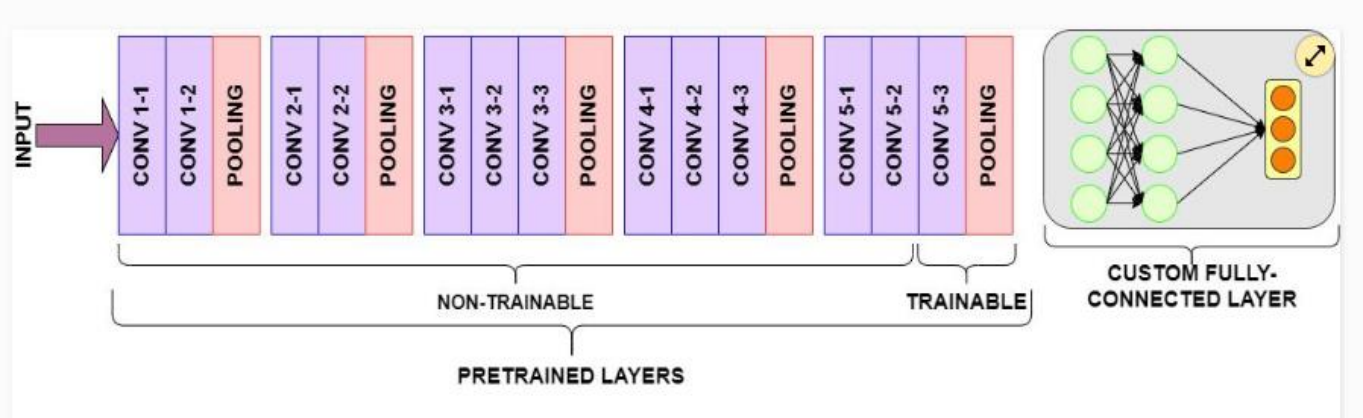
1. Feature Extraction Approach

In this technique, we leverage the design of the pre-trained models to construct a new dataset from the inserted pictures. The Convolutional & Pooling layers are going to be put in, but the "upper section" of the cnn model will remain out (the completely Connected layer). We already knew that VGG16 was trained on millions of photos, even having photographs of vehicles. Its taught weights & convolutional layers can recognise general characteristics like edges, colours, wheels, and windshields. We will run photos through VGGsixteen’s convolutional layers, which will be going to provide a Feature Stack containing the “visual features” that were discovered. It becomes simple to flat the 3D feature stack into a NumPy array from here, making it ready for any modelling user want to do.



2. Fine Tuning Approach

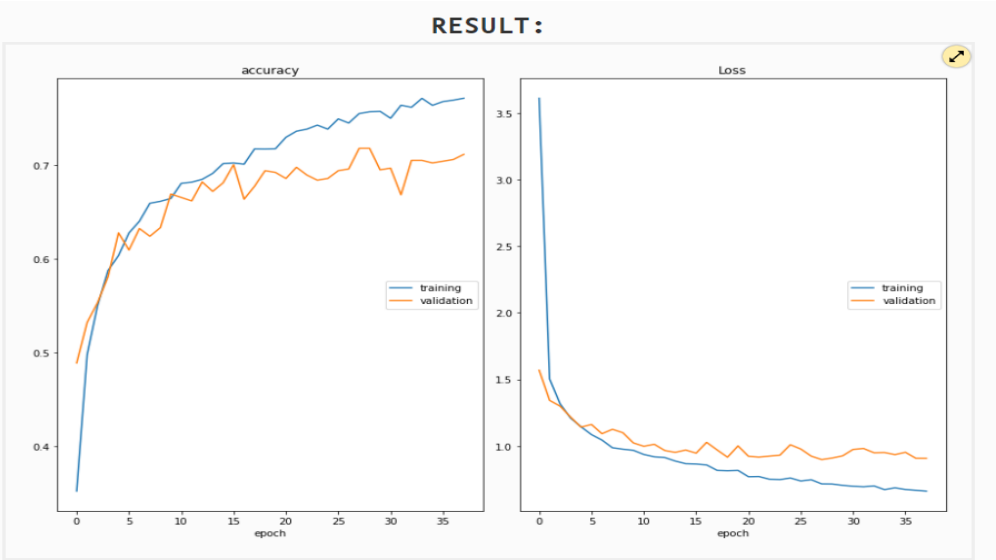
The main purpose of fine tuning is to let some of previously learned levels of layers to get relearnt. In order to extract features in prior method, they have employed VGG16's pre-trained layers. The altered visual characteristics were produced after passing our picture dataset through the convolutional layers andparameters. These pre-trained layers



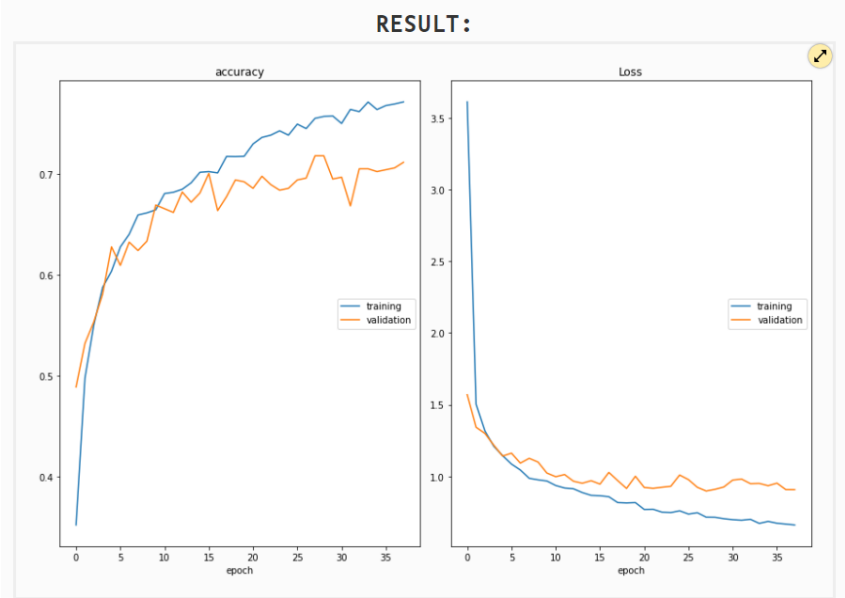
received no genuine teaching.

Using Pre-trained Layers for Feature Extraction

->Training Without Fine-Tuning



->Training Without Fine-Tuning



The best system is transfer learning with fine-tuning, as seen by the stronger diagonal and lighter cells everywhere else . The confusion matrix indicates that the model frequently misclassifies apple pie as bread pudding.

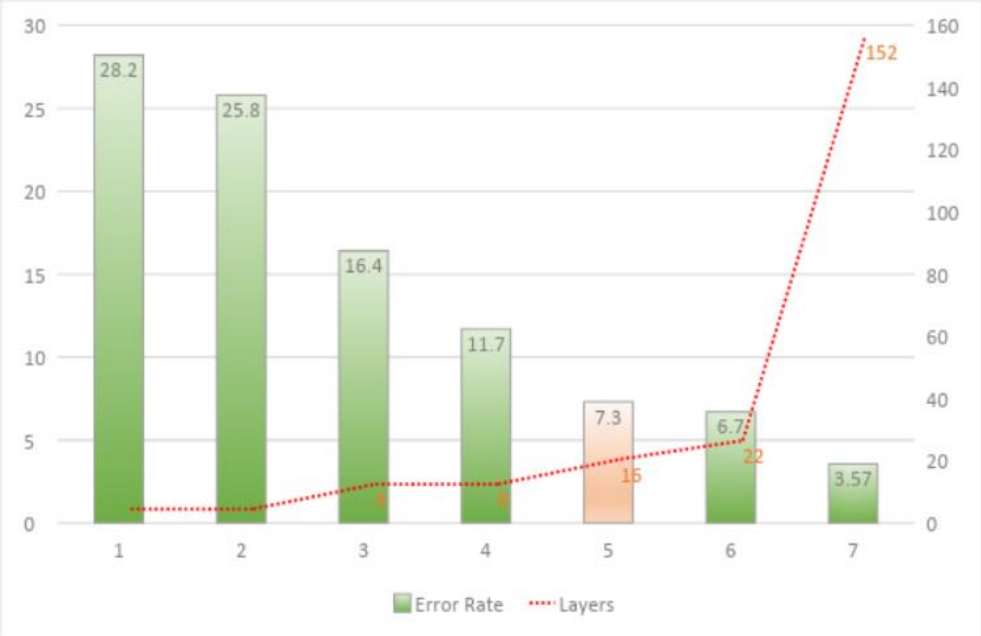


Fig 1: ImageNet Results: VGG Vs Others

Despite the fact that VGG16 did not even win the ImageNet 2014 Challenge, the ideas it presented laid the ground for later advancement in the area of computer vision. In my perspective, the concept of layered convolution layers with smaller receptive fields was the one who changed the whole game.It also provides some of the best advantages in comparison to the previous network. This block of stack convolutions is even now used by most of the cnn models in the network.

3.6 Xception

The Xception Methodology is developed by Francois Chollet.Xception is an augmentation of the Cnn Architectures that replaces the conventional Inception modules with depthwise Separable Convolutions.

Xception is a seventy-one-layer deep neural network. The ImageNet dataset comprises cnn model network that has been pretrained on a large number of pictures. The network was able to classify pictures into thousand different item categories, including keyboards , mouse , pencils , and other animals. As a result, the network has learned a wide range of rich characterstic representations for a wide range of images. The image data size for this network is 299 * 299 pixels.
With basic stacking of convolutional layers for extracting features and max-pooling layers for spatially subsampling in the LeNet-style , AlexNet , and Vgg models, to Inception as well as ResNet networks, that employ hidden layers & numerous convolutional as well as max-pooling chunks inside each layer. Since its beginnings, the Inception network has grown to be one of the most important networks in machine learning. The Inception model stacks modules, which each includes based feature feature extraction methods that let it to learn stronger presentations with lower complexity.

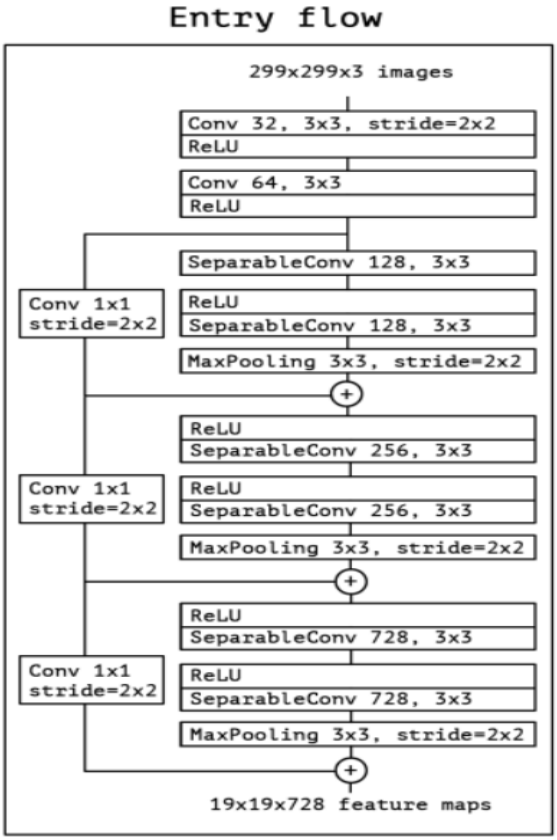


Figure 8: Entry Entry flow in Xception CNN

Two convolution layers block follows a ReLU activation in the entering flow. The number of filters, filter size (kernel size), and strides are all detailed in the figure. Separable convolutional layers also are available. Max Pooling layers are also available. When there are more than one step, the strides are indicated as well. There are also Skip connections, where the 2 tensors are combined using 'ADD'. In each cycle, this also illustrates the form of the given tensor. For example, we start with a 299x299x3 image and end up with a 19x19x728 image following the entrance flow.

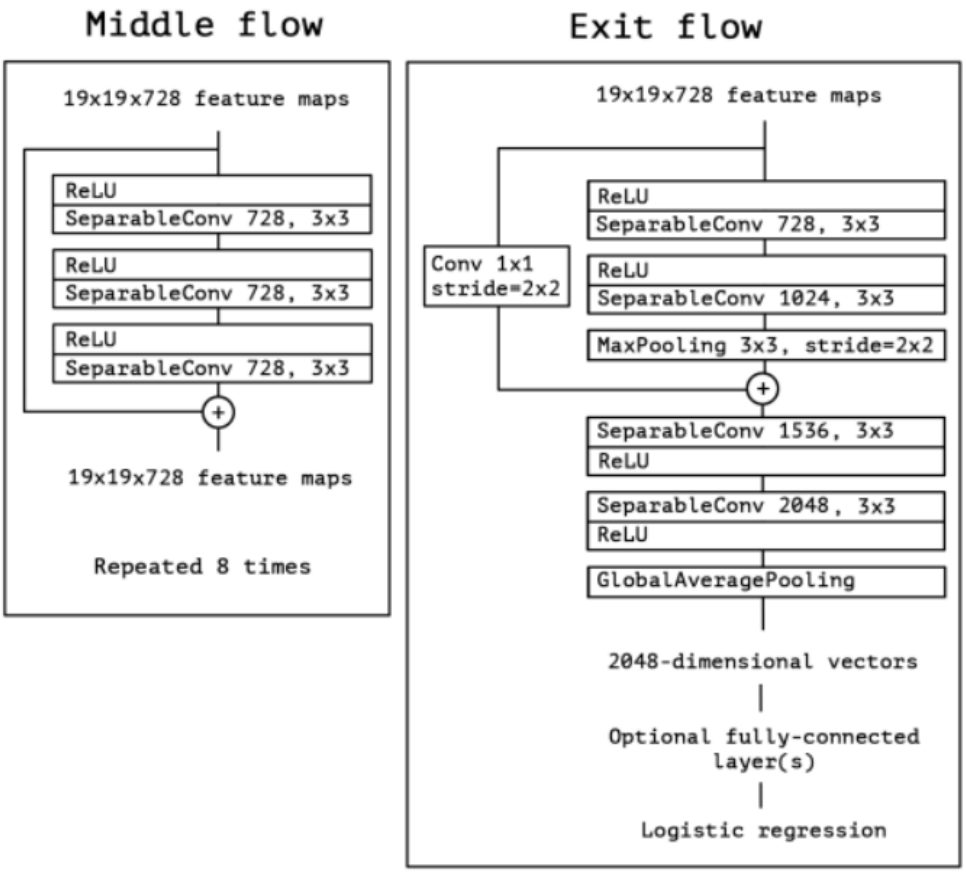


Figure 9: Middle and Exit flow of Xception CNN

Similarly, for the Center & Departure stages, the figure clearly describes the file sizes, numerous layers, quantity of filters , circuit design, type of pool, quantity of repeats, as well as the possibility of inserting a completely connected layer at the conclusion.

VGGNet – 1st Runner Up in ILSVRC 2014

ResNet – Winner in ILSVRC 2015

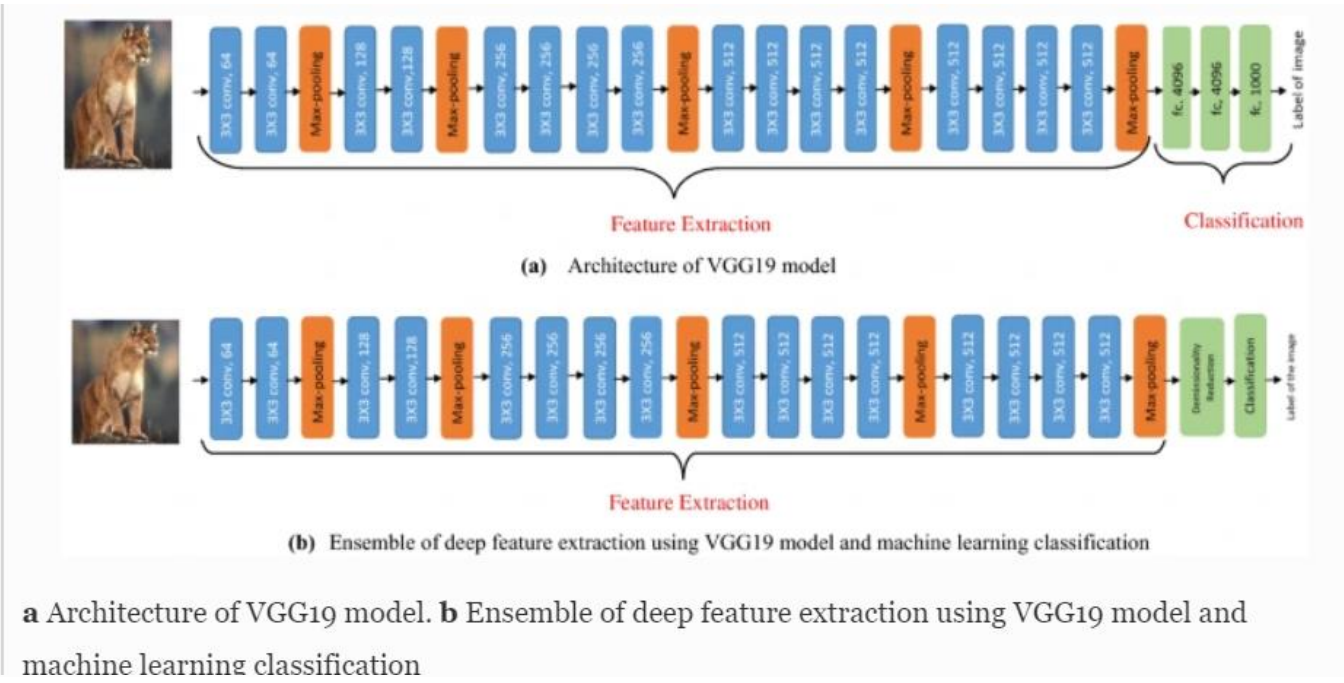
Inception-v3 – 1st Runner Up in ILSVRC 2015

	Top-1 accuracy	Top-5 accuracy
VGG-16	0.715	0.901
ResNet-152	0.770	0.933
Inception V3	0.782	0.941
Xception	**0.790**	**0.945**

ImageNet: Xception has the Highest Accuracy

3.7 VGG19

VGG19 is a convolutional neural network introduced by Simonyan and Zisserman with Nineteen layers in which there are Sixteen convolution layers and the three fully connected layers for classifying pictures into thousand item categories. VGG19 was educated to use the ImageNet dataset, which has a million pictures divided into thousand categories. It is a particularly prevalent strategy for image standard list to the use of many 3 3 filters in each convolutional layer. The design of VGG19 may be observed in the diagram. The very first Sixteen convolutional layers are required for extracting features, so the next three layers are being used for categorization. The extraction characteristics layers are divided into five categories, each with its own maxpooling layer. This technique takes a 224 * 224 pixel picture & generates the label for the object in the picture. The research uses a pre-trained VGG19 model to extract information, however other ML approaches are used for categorization. Because the Cnn architecture estimates high parameters after extracting features, as shown in Fig, data reduction is essential to keep the feature representation short. To minimise complexity, a Locality Preserving Projection is utilised, followed by a classification algorithm.



a Architecture of VGG19 model. **b** Ensemble of deep feature extraction using VGG19 model and machine learning classification

VGG19 is a powerful CNN with already layers as well as a deep understanding of how an image is characterized in terms of shape, colors, & structure. VGG19 is a deep neural network learned on millions of pictures with difficult classification tasks.VGG19 was not additionally trained; instead, its layers were frozen and a shallow two-layer network was placed on top of that to accomplish my categorization job of distinguishing photos with and without trees.

VGG19 has been the most latest VGG model, so it looks very similar to VGG16. When you examine the structure of the model beneath to those of VGG16, you'll notice that they're both built on five convolutional blocks. But, by adding a convolutional layer in the last three blocks, the network's complexity has been boosted even further. The intake is indeed an RGB picture with the form (224,224,3), and the output is a features tensor with the shape (224,224,3). (7,7,512). VGG19 has its own preprocessing method in Keras, but if you examine at the coding, you'll notice that it's exactly somewhat like VGG 16. As a result, we won't have to rewrite the database iterators.

The vgg19 model is performed by congiguring the following steps:

1) Loading the model.

```
# Build VGG16 structure
cnn_base = VGG19(weights='imagenet',
                 include_top=False,
                 input_shape=(228, 228, 3))
print('VGG19 Loaded')
print(cnn_base.summary())
```

2) Load the data set size.

```
# Specify dataset size
batch_size = 16
nb_train_samples = 2084
nb_validation_samples = 167
nb_test_samples = len(os.listdir('data/test/tree')) + len(os.listdir('data/test/not_tree'))
```

3)Create a method to retreive and hold the VGG-19's layer which behind a hood involved in the production feature representation. This one will enable the model to use transfer learning, in which it will be able to recollect its alreadt examined with millions of photographs available at internet.

```
def extract_features(directory, sample_amount):
    features = np.zeros(shape=(sample_amount, 7, 7, 512))
    labels = np.zeros(shape=(sample_amount))
    datagen = ImageDataGenerator(rescale=1./255)
    generator = datagen.flow_from_directory(
        directory, target_size=(228, 228),
        batch_size = batch_size,
        class_mode='binary')

    i = 0
    for inputs_batch, labels_batch in generator:
        features_batch = cnn_base.predict(inputs_batch)
        features[i * batch_size : (i + 1) * batch_size] = features_batch
        labels[i * batch_size : (i + 1) * batch_size] = labels_batch
        i = i + 1
        if i * batch_size >= sample_amount:
            break
    return features, labels
```

4) Use the method to retreive the information & tags from the training data, verification data, and testing data.

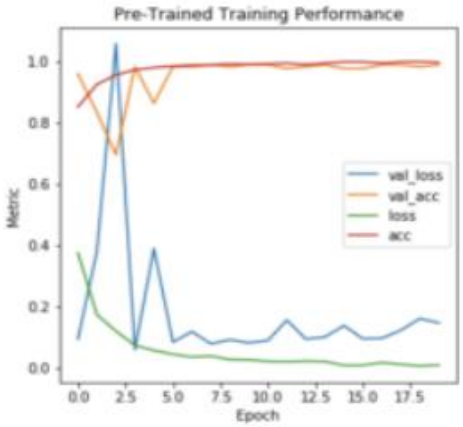
```
# Apply extraction function to 3 datasets
train_features, train_labels = extract_features(train_folder, nb_train_samples)
validation_features, validation_labels = extract_features(val_folder, nb_validation_samples
test_features, test_labels = extract_features(test_folder, nb_test_samples)
```

5) Save the features that are extracted and labels inside a 'artificially limited' directory for your lastly categorizating layer to belong to when determining which binary group a picture that it refers to.

6) Create the last layer of the picture classifier on top of the "VGG-10 brain" & set it to perform some action.

7)vGenerate your training record to see how your model learnt using your accuracy and loss measures. As you'll see, our accuracy and losses dropped with each epoch (or repetition)

```
In [31]: # Print training history
         pd.DataFrame(history.history).plot(figsize=(5, 5))
         plt.title('Pre-Trained Training Performance')
         plt.xlabel('Epoch')
         plt.ylabel('Metric')
         plt.show()
```



3.8 NASNetMobile

The Google brain group designed the NASNetMobile "Neural Architecture Search Network" (Nasnet), that employs 2 significant characterstics one is Normal cell and other is reduction cell. Nasnet first performs its activities on a tiny database before transferring its chunk to a huge database to get a greater mAP. Nasnet's speed is increased by using a customized droppath termed Scheduled droppath for better regularisation .In the initial Nasnet Design, the cells is not pre-determined, but only normal & reduction cells are utilized. Normal cells determine the feature map dimensions, while reduction cells yield an extracted features that has been reduced by a factor of 2 in length & breadth. Based on 2 initial random variables, a Nasnet controlling designed to look on Recurrent Neural Network forcasts the whole topology of the network.

The cnn NASNet Mobile was developed on over a thousand photos as from Imagenet collection. The network can sort photos into thousand distinct object classes, including keyboards, mice, pencils, as well as a variety of creatures. As a consequence , the networks has learnt a variety of rich image features for a variety of pictures. The network's picture element size is 224 × 224 pixels.

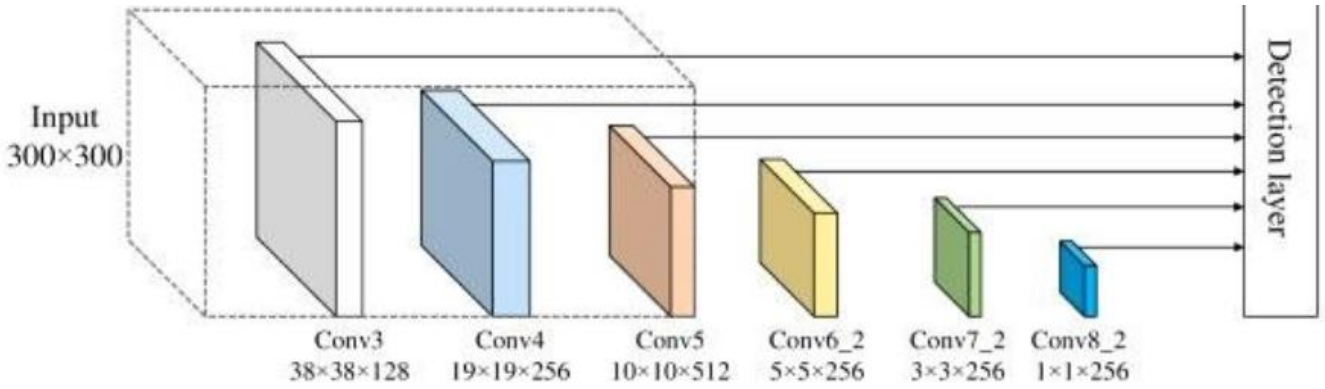
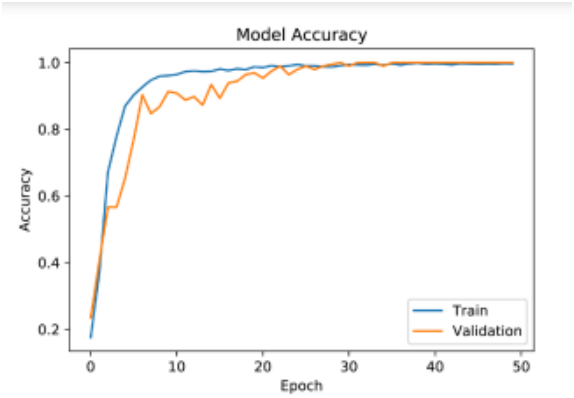


Figure 10: NASNetMobile CNN Architecture

Google announced NASNet , that presented the task of choosing the desired Cnn model as a Reinforcement Learning problem, kudos to its vast computer capacity & technological talent.

Its basic concept was always to find the optimal mixture of window sizes, perspectives show, durations, amount of hidden layers, and other characteristics in the specified search area. The correctness of the found design on the provided database was the incentive for every searching operation in this Reinforcement Learning environment.

Inside the ImageNet competition, NASNet received a state of the art score. Just a few organisations was able to adopt the very same concept since the computation power needed for NASNet was just so large.



NASNetMobile

Two different types of convolutional cells are employed several times in NASNetMobile. There are two types of cells: normal and reduced. Both of these methods provide a feature map.

The goal of the NASNet study was to use reinforcement learning to find the best CNN architecture. NAS refers for Neural Architecture Search and is a method for searching across a space of neural network configurations created at Google Brain. To optimise CNNs for different sizes, NAS was employed with standard datasets as CIFAR10 and ImageNet. NASNetMobile is the abbreviated version. The best reduced convolutional cell produced using NAS and CIFAR10 is shown below.

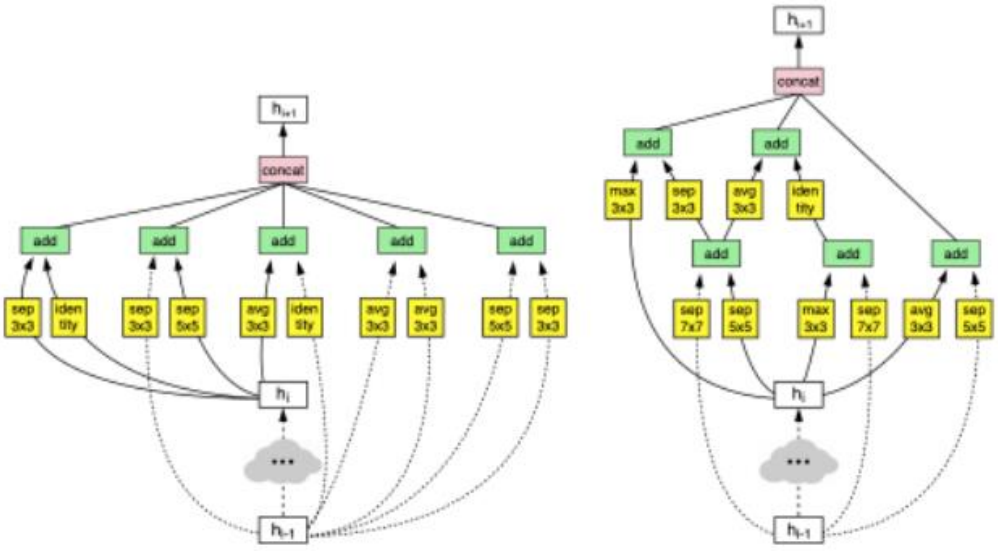


Figure 4. Architecture of the best convolutional cells (NASNet-A) with $B = 5$ blocks identified with CIFAR-10 . The input (white) is the hidden state from previous activations (or input image). The output (pink) is the result of a concatenation operation across all resulting branches. Each convolutional cell is the result of B blocks. A single block is corresponds to two primitive operations (yellow) and a combination operation (green). Note that colors correspond to operations in Figure 3.

Chapter 4 : Performance Analysis

Four unique alredy trained CNN models are tested on a dataset to accomplish 3 binary classification tasks, including lung cancer subgroup categorization, detecting malignant and benign colon, and lung histology picture analysis. For testing predictions, which account for 20percent of the entire dataset, test data is used. Model effectiveness is evaluated using precision, recall, F1-score , accuracy, and Auroc rating. To gain a visual examination of model performance during learning the testing & validation information, training loss, validation loss , training accuracy , and validation accuracy with respect to epochs are also presented. The validation data utilised accounts for 20percentage points of the total training data (i.e., after 80:20 train test split).

4.1 Lung cancer

Table lists the evaluation metrics for each of the four models. All 4 Pretrained CNN architectures got 100percent precision, accuracy, recall, f1score, and auroc scores, according to expectations. However, there is one caveat: the NASNetMobile model has a 99 percent auroc score.

Table 2: Evaluatuion of Four pretrained CNN models identifying the Lung Cancer

Model	Precision	Recall	F1-score	Accuracy	Auroc
VGG16	1.0	1.0	1.0	1.0	1.0
Xception	1.0	1.0	1.0	1.0	1.0
NASNetMobile	1.0	1.0	1.0	1.0	0.99
VGG19	1.0	1.0	1.0	1.0	1.0

Table lists the evaluation metrics for all four models. All four Pre-trained CNN models achieved excellent outcomes in precision, accuracy, recall, f1score, and auroc rating, according to forecasts. However, there still is one caveat: the NASNetMobile model has a 99 percent auroc rating.

Figure indicates that, with the exception of ResNet50, all models indicate similar graphs, showing early convergence. ResNet50 exhibited a dramatic increase in reducing losses & optimising correctness in the last 2-3 epochs.

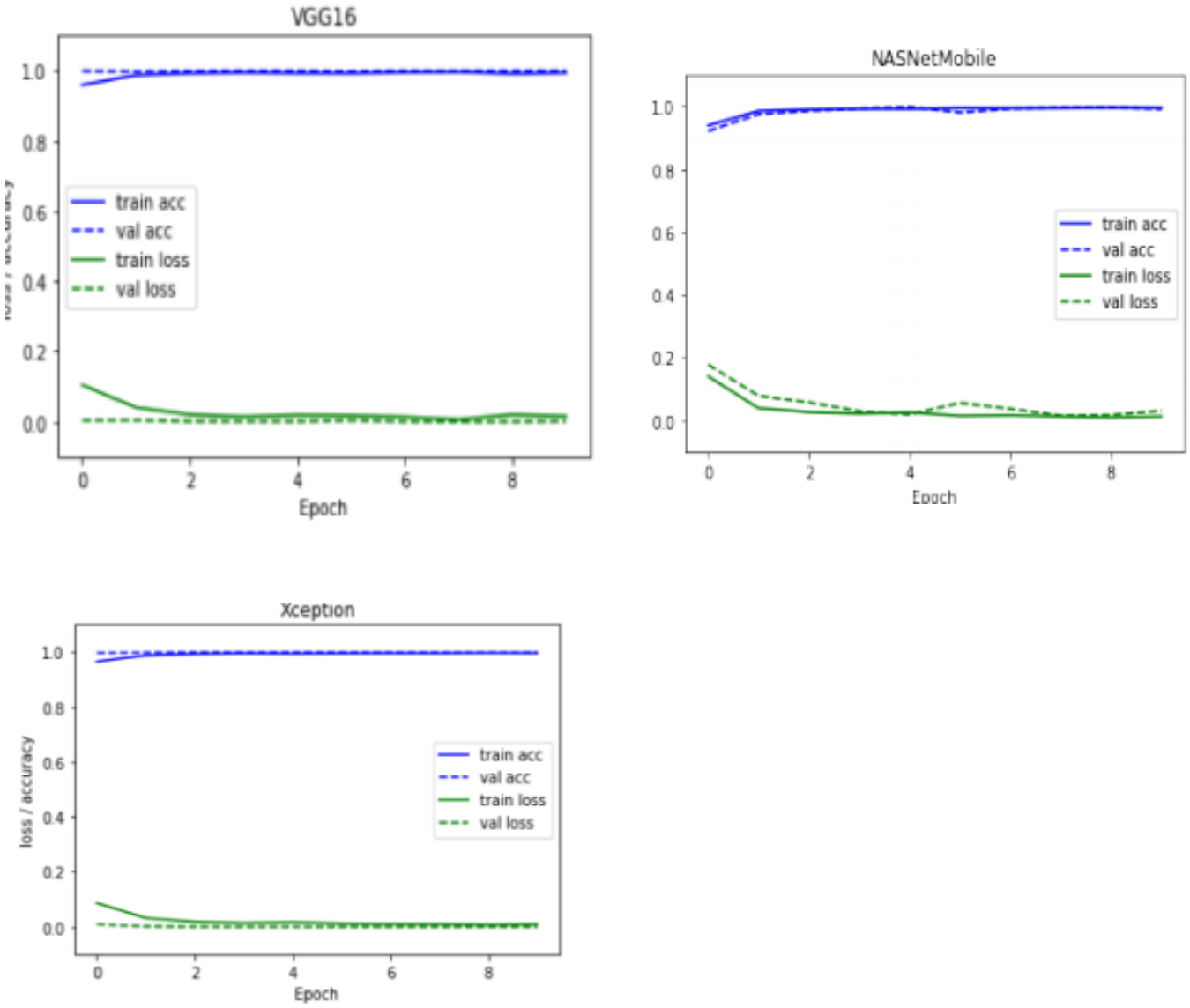


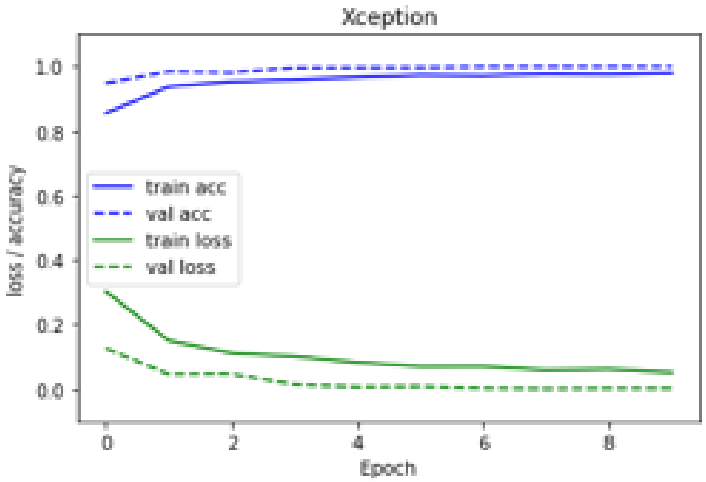
Figure 11: Change in loss and accuracy of training and validation data with respect to the number of 4 pretrained CNN classifying Lung images.

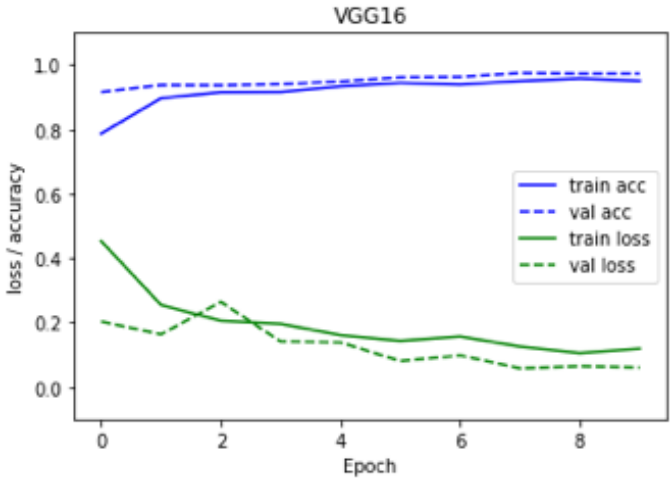
4.2 Lung cancer subtypes

Lung cancer subgroups are classified as lung carcinoma and lung squamous cell carcinoma. Table shows that VGG19 and Xception also scored 100 percent across all assessment criteria except the auroc score, which was achieved by NASNetMobile at 99.9percent. The other models, the VGG16, Resnet50, and NASNetMobile , scored 96percent to 98percent.

Table 3. Evaluation of eight Pre-trained CNN models identifying type of lung cancer

Model	Precision	Recall	F1-score	Accuracy	Auroc
VGG16	0.975	0.975	0.98	0.98	0.999
Xception	1.0	1.0	1.0	1.0	1.0
NASNetMobile	1.0	1.0	1.0	1.0	1.0
VGG19	0.965	0.965	0.97	0.97	0.997





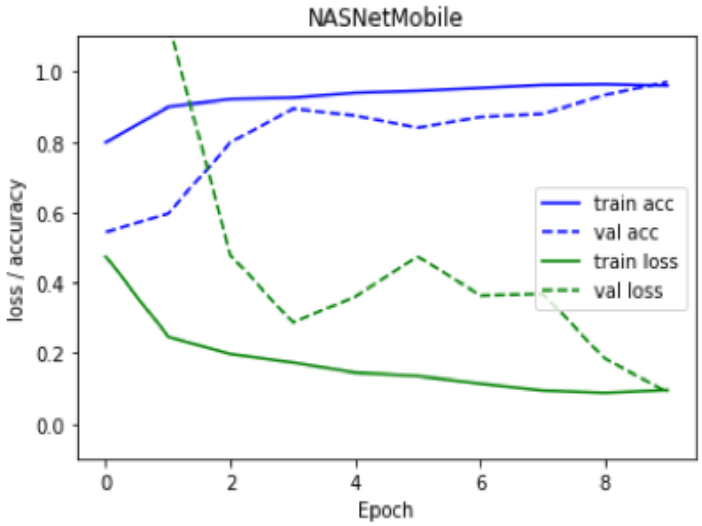


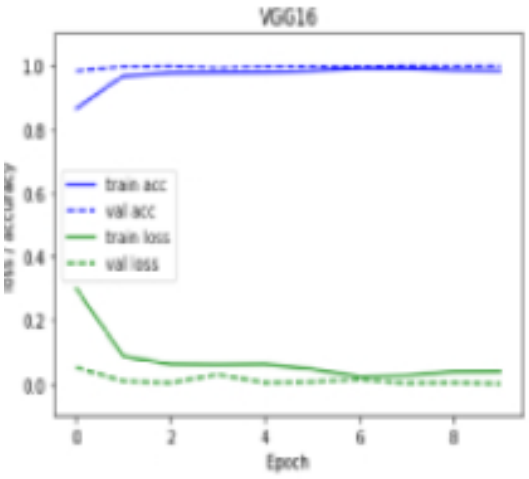
Figure 12: Change in loss and accuracy of training and validation data with respect to the number of 4 pretrained CNN classifying Lung cancer subtypes images.

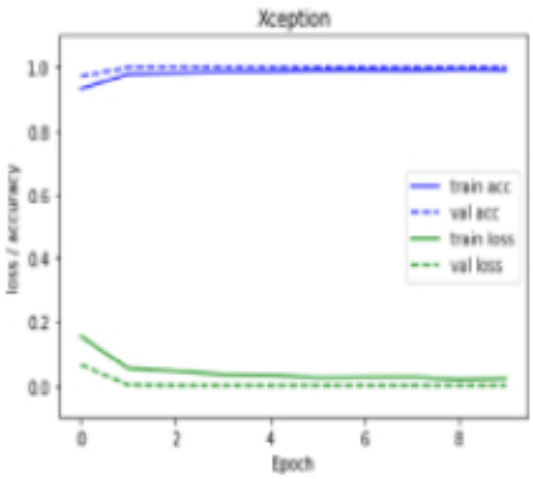
The green validation loss curve(dotted) is somewhat is toward upper end, indicating that NASNetMobile is not in agreement with other models. Furthermore, ResNet50 displayed confusion in classifying lung pictures, following a similar trend.

4.3 Colon Cancer

Table summarises how well the models scored, with 100 percent accuracy throughout all assessment measures. NASNetMobile , while moving to second phase,achieved approximately 98percent accuracy. These are, for the most part, excellent results. Figure shows that ResNet50 produced a curve similar to the one shown in the lung classification model. The zigzag structure of the validation loss & accuracy curve of the NASNetMobile design depicts the 2percent decline in evaluation metrics of NASNetMobile .

Model	Precision	Recall	F1-score	Accuracy	Auroc
VGG16	1.O	1.0	1.0	1.0	1.0
Xception	1.0	1.0	1.0	1.0	1.0
NASNetMobile	1.0	1.0	1.0	1.0	1.0
VGG19	1.0	1.0	1.0	1.0	1.0





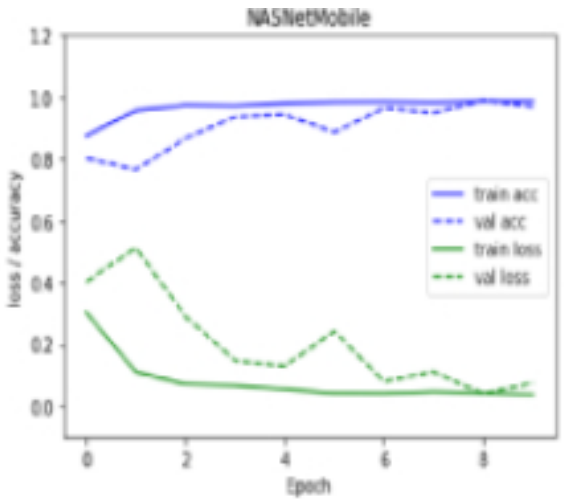


Figure 13: Change in loss and accuracy of training and validation data with respect to the number of epochs of four Pre-trained CNN models classifying the colon cancer.

Chapter 5: Conclusions

5.1 Conclusion

Lung cancer is one of the most deadly illnesses that has ever existed. However, once the disease has grown or reached a dangerous level, it is exceedingly harder to cure. Computer-Aided Detection (CAD) is a rapidly evolving technology that aids in the detection of cancer by putting in patient-related data such as CT scans, X-rays, Mris, odd symptoms in individuals, biomarkers, and so on. SVM , CNN, ANN, Watershed Segmentation, Image enhancement and picture analysis are some of the techniques being used to increase accuracy & speed up the process. The most often used data for training are LUNA16 , "Super Bowl Dataset" 2016, and "LIDC-IDRI". We hope that by completing this review article, we will be able to identify all of the key studies that have been conducted in recent years and can be improved upon to reach better outcomes.We emphasised the necessity for automation in clinical imaging in this study, which can drastically reduce costs specialists to reduce the heap Then we show comparable works that range from simple to comple help advance models such as deep learning On our histopathological images, we tried out different information expansion algorithms at that moment. In all classification, we used four famous already trained CNN models: VGG16 ,VGG19 , NASNetMobile , and Xception , which achieved outstanding results varying from 97.5percent to 100percent accuracy . In particular, MobileNet achieved 100percent precision, accuracy, auroc, score , recall , and f1-score .Models other than these models have likewise achieved outstanding score which is ranging from 97.5percent to 99.5percent. Such huge levels of accuracyy are the consequence of mixing a complicated picture data increasing pipeline (imgaug) with usage of already trained CNN models. Development & validating damage & efficiency of numerous already trained CNN models have previously been imagined, in additional to the model's evaluation layers. As in long term, we will combine diverse databases to use all of the stated methodologies to detect if the images have more cancer types.

5.2 Future Work

The future goal of this study is to improve on the present already examined CNN based which was used for predicting cancer using the images present in the dataset.The LC25000 dataset is used to train more different pre-trained CNN models:InceptionV3, ResNet50and DenseNet169.

5.2.1 ResNet50

ResNet50 is a deep neural model with Fifty layers. We can load a already trained variant of the network which have been already learnt on over a lot of photographs using the ImageNet database.The achitecture was able to classify pictures into thousands distinct item categories, included keyboards , mouse , pencils , and other creatures. As a result, the system has learned a wide range of rich feature representations for a wide range of images. The digital image size for the system is 224 * 224 pixels.

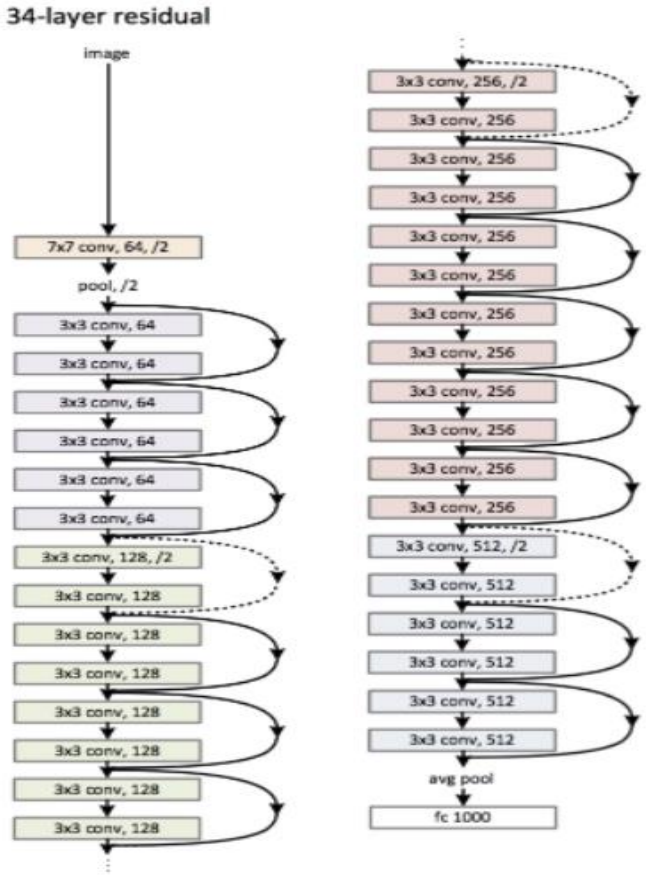


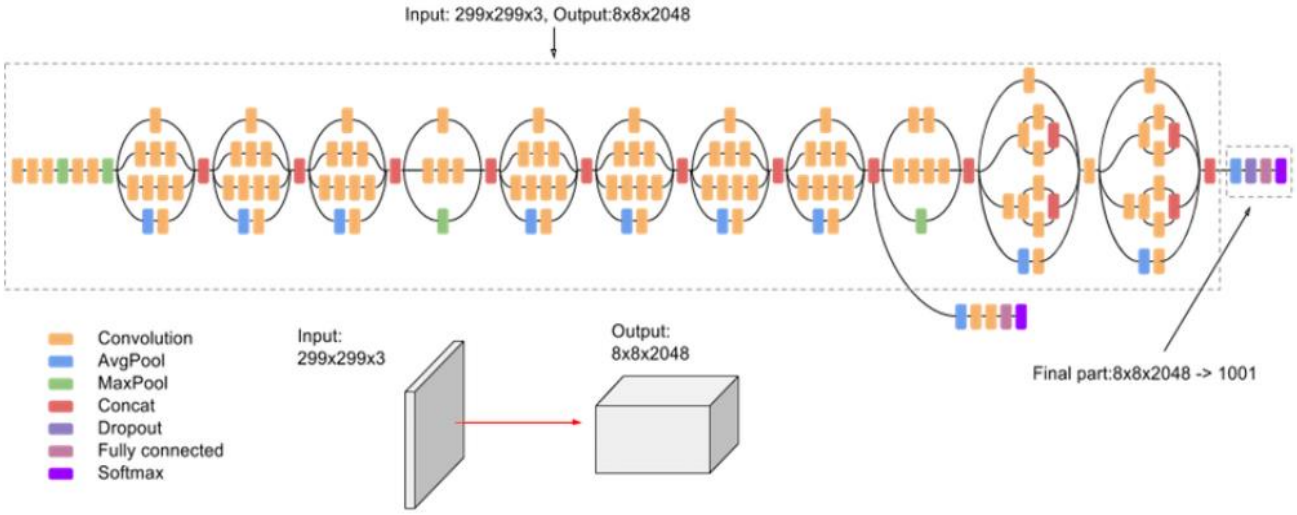
Figure 14: Resnet50 Architecture

This design could be used for computer vision tasks such as picture categorization, item localization, and object identification, but it can also be utilised for non-computer vision operations to provide depth & minimise processing costs.

5.2.2 InceptionV3

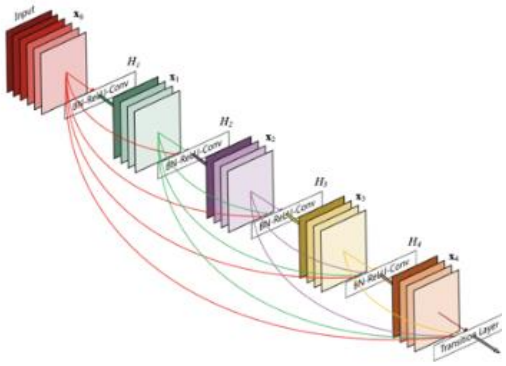
Inception v3 is a Google net element that's been created to assist in image analysis & object identification. It's the 3rd iteration of Google's “Inception Neural Network”, originally debuted at the "ImageNet Recognition Challenge" last year. Inceptionv3 is designed to allow for deeper networks while preserving effect on the number of parameters: it contains "under 25 million variables," compared to AlexNet's sixety million.

On the Imagenet database, Inception v3 is an picture identification model which has been demonstrated to having more than 78.1percent efficiency. The concept is the culmination of numerous notions developed over the years by a number of academics. It is based on the study "Rethinking the Inception Architecture for Computer Vision" by Szegedy et al". The model's balanced and unbalanced construction components include convolution layer, pooling layers, max pooling, interlinking, failures, and completely connected levels. Batch normalisation is applied to activating variables and is used throughout the model. Loss can be calculated using Nonlinear function.



5.2.3 DenseNet169

DenseNet is a novel neural network for image classification and object that was discovered recently. DenseNet and ResNet are quite comparable, however There seem to be a few significant differences. ResNet uses an addition technique that merges the previous layer with both the future stacking, whereas DenseNet combines the outcome of the preceding phase with the output of another level.
DenseNet was developed to address the fading gradients in high level neural nets, that leads to a loss of efficiency. To put it differently, due to the lengthier route here between input and output layers, data disappears before it gets its goal.
The main goal of this tutorial is to teach you how to use TensorFlow 2.0 (TF 2.0) and Keras to develop DenseNet121.



DenseNet Structure

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

Figure 15: DenseNet Architecture

The vanishing gradient problem is reduced using DenseNet, as well as the model may be trained with less variables. The smooth flow of data is ensured via dynamic feature propagation.
This will covers the fundamentals of developing the DenseNet-121, including its design, benefits, and differences from ResNet. If in a figure We observe from the heat map that dogs have been misclassified as cats, probably because the misclassified dog photos resemble cats. The model may be fine-tuned to improve the results. Adding or eliminating more dense blocks and layers, determining the frequency of data in each class, and supplementing the photos are all possibilities.

5.2.4 MobileNet

TensorFlow's initial mobile computer visual model is the MobileNet model, which is geared for usage in mobile applications. MobileNet employs feature maps convolutions . As contrasted to a system with traditional convolution layers with similar depths in the network, the complexity is substantially reduced. As a result, light neural networks are efficiently developed.
Depthwise convolution and Pointwise convolution are the two procedures that make up a depthwise separable convolution.
MobileNet is a cnn category which Google open sourced, so it gives us a great opportunity to begin building a small and compact & rapid classification.

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Visualizing activation maps aids in highlighting crucial sections of the picture that activation is strong. It offers users confidence that their work is proceeding in the right direction. GradCam and SmoothGrad are used in this work to show class activation and saliency maps. Saliency charts like SmoothGrad and vanilla saliency are using the gradients of a output nodes with respect to the input image to tell us how the resulting value changes as the input data keeps changing, while a class activation maps like GradCAM, ScoreCAM, and others don't use gradients and use the penultimate layer to get spatial information that gets lost in dense layers. These maps were visualised using the tf-keras-vis visualisation toolbox.
Figure illustrates gradient-based class activation maps & saliency maps created with GradCAM and SmoothGrad to visualise focus using the NASNetMobile lung model and the VGG16 colon model to categorise malignant and noncancerous pictures.

5.2.5 GradCAM

Gradcam employs the objective feature's parameters, that flow in to final convolutional layer, producing in a width as well as height localization map for any subcategory. It also necessitates the estimation of neuron significance parameters, which can be done by comparing the gradient of a specific class during softmax () to the feature map activation () of a convolution layers.

$$\alpha_k^c = \frac{1}{Z} \Sigma_i \Sigma_j \frac{\partial y^c}{\partial A_{ij}^k}$$

$$L_{Grad-CAM}^c = ReLU\left(\Sigma_k \ \alpha_k^c A^k\right)$$

5.2.5 SmoothGrad

Vanilla saliency or SmoothGrad can be used to visualise saliency maps. Because the vanilla saliency was very much noisy, SmoothGrad was included in this study. The

SmoothGrad adds noise to the input picture in order to improve the saliency maps. The final classification () for pictures after predicting a class activation function for every category is:

$$class(x) = argmax_{c \in C}\ S_c(x) \tag{8}$$

Sensitivity map $M_c(x)$ can be calculated by differentiating S_c with respect to input x.

$$M_c(x) = \partial S_c(x)/\partial x \tag{9}$$

It has been shown that the gradient of S_c is producing fluctuations rapidly [9]. To improve sensitivity maps, a neighborhood average of gradient values has been used on smoothing of ∂S_c with a Gaussian kernel. The smoothed gradient $\hat{M_c}(x)$ for an input image x, is represented by:

$$\hat{M_c}(x) = 1/n\ \Sigma_i^n M_c(x + N(0, \sigma^2)) \tag{10}$$

where n is the number of samples, $N(0, \sigma^2)$ is Gaussian noise with standard deviation σ and M_c represents unsmoothed gradient [9].

References

[1] Cancer Key Facts, **https://www.who.int/news-room/fact-sheets/detail/cancer** Last accessed 25 Aug 2020

[2] Komura, Daisuke & Ishikawa, Shumpei. (2017). Machine Learning Methods for Histopathological Image Analysis. Computational and Structural Biotechnology Journal. 16. 10.1016/j.csbj.2018.01.001.

[3] Dela Cruz, C. S., Tanoue, L. T., & Matthay, R. A. (2011). Lung cancer: epidemiology, etiology, and prevention. Clinics in chest medicine, 32(4), 605–644. **https://doi.org/10.1016/j.ccm.2011.09.001**

[4] Rawla, P., Sunkara, T., & Barsouk, A. (2019). Epidemiology of colorectal cancer: incidence, mortality, survival, and risk factors. Przeglad gastroenterologiczny, 14(2), 89–103. **https://doi.org/10.5114/pg.2018.81072**

[5] Jung, A. (2019). Imgaug documentation. Readthedocs. io, Jun, 25.

[6] Krizhevsky, Alex & Sutskever, Ilya & Hinton, Geoffrey. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Neural Information Processing Systems. 25. 10.1145/3065386.

[7] Powers, David & Ailab,. (2011). Evaluation: From precision, recall and F-measure to ROC, informedness, markedness & correlation. J. Mach. Learn. Technol. 2. 2229-3981. 10.9735/2229-3981.

[8] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, 2016; arXiv:1610.02391. DOI: 10.1007/s11263-019-01228-7.

[9] Daniel Smilkov, Nikhil Thorat, Been Kim, Fernanda Viégas and Martin Wattenberg. SmoothGrad: removing noise by adding noise, 2017; arXiv:1706.03825.

[10] Pooya Mobadersany, Safoora Yousefi, Mohamed Amgad, David A. Gutman, Jill S. Barnholtz-Sloan, José E. Velázquez Vega, Daniel J. Brat, Lee A. D. Cooper. Proceedings of the National Academy of Sciences Mar 2018, 115 (13) E2970-E2979; DOI: 10.1073/pnas.1717139115

[11] Jiao, Liping & Chen, Qi & Li, Shuyu & Xu, Yan. (2013). Colon Cancer Detection Using Whole Slide Histopathological Images. IFMBE Proceedings. 39. 1283-1286. 10.1007/978-3-642-29305-4_336.

[12] Evgeniou, Theodoros and Pontil, Massimiliano. (2001). Support Vector Machines: Theory and Applications. 2049. 249-257. 10.1007/3-540-44673-7_12.

[13] S. Doyle, S. Agner, A. Madabhushi, M. Feldman, and J. Tomaszewski, “Automated grading of breast cancer histopathology using spectral clustering with textural and architectural image features,” in 2008 5th IEEE International Symposium on Biomedical Imaging: From Nano to Macro, Proceedings, ISBI, 2008.

[14] S. Rathore, M. Hussain, and A. Khan, “Automated colon cancer detection using hybrid of novel geometric features and some traditional features,” Comput. Biol. Med., 2015.

[15] Hawkins, Douglas. (2004). The Problem of Overfitting. Journal of chemical information and computer sciences. 44. 1-12. 10.1021/ci0342472.

[16] F. A. Spanhol, L. S. Oliveira, C. Petitjean, and L. Heutte, “Breast cancer histopathological image classification using Convolutional Neural Networks,” Int. Jt. Conf. Neural Networks, 2016.

[17] Andrew A. Borkowski, Marilyn M. Bui, L. Brannon Thomas, Catherine P. Wilson, Lauren A. DeLand and Stephen M. Mastorides. Lung and Colon Cancer Histopathological Image Dataset (LC25000), 2019; arXiv:1912.12142.

[18] [1] Spanhol, F., Oliveira, L. S., Petitjean, C., Heutte, L., A Dataset for Breast Cancer Histopathological Image Classification, IEEE Transactions on Biomedical Engineering (TBME), 63(7):1455-1462, 2016.
[19] Hiba Chougrad, Hamid Zouaki and Omar Alheyane. Convolutional Neural Networks for Breast Cancer Screening: Transfer Learning with Exponential Decay, 2017; arXiv:1711.10752.
[20] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision, 2015; arXiv:1512.00567.
[21] Rebecca Sawyer Lee, Francisco Gimenez, Assaf Hoogi, Kanae Kawai Miyake, Mia Gorovoy and Daniel L. Rubin. (2017) A curated mammography data set for use in computer-aided detection and diagnosis research. Scientific Data volume 4, Article number: 170177 DOI: https://doi.org/10.1038/sdata.2017.177
[22] Moreira, I. C., Amaral, I., Domingues, I., Cardoso, A., Cardoso, M. J., & Cardoso, J. S. (2012). INbreast: toward a full-field digital mammographic database. Academic radiology, 19(2), 236–248. https://doi.org/10.1016/j.acra.2011.09.014
[23] Guevara Lopez, Miguel Angel & Posada, Naimy & Moura, Daniel & Pollán, Raúl & Franco-Valiente, José & Ortega, César & Del Solar, Manuel & Díaz-Herrero, Guillermo & Ramos, Isabel & Loureiro, Joana & Fernandes, Teresa & Araújo, Bruno. (2012). BCDR: A BREAST CANCER DIGITAL REPOSITORY. 1065-1066.
[24] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Aude Oliva and Antonio Torralba. Learning Deep Features for Discriminative Localization, 2015; arXiv:1512.04150.
[25] Karen Simonyan, Andrea Vedaldi and Andrew Zisserman. Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps, 2013; arXiv:1312.6034.
[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. Deep Residual Learning for Image Recognition, 2015; arXiv:1512.03385.
[27] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization, 2016; arXiv:1610.02391. DOI: 10.1007/s11263-019-01228-7.
[28] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, 2014; arXiv:1409.1556.
[29] Howard, Andrew & Zhu, Menglong & Chen, Bo & Kalenichenko, Dmitry & Wang, Weijun & Weyand, Tobias & Andreetto, Marco & Adam, Hartwig. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.
[30] Gao Huang, Zhuang Liu, Laurens van der Maaten and Kilian Q. Weinberger. Densely Connected Convolutional Networks, 2016; arXiv:1608.06993.
[31] Elgendy, Nada & Elragal, Ahmed. (2014). Big Data Analytics: A Literature Review Paper. Lecture Notes in Computer Science. 8557. 214-227. 10.1007/978-3-319-08976-8_16.
[32] Aresta, G., Araújo, T., Kwok, S., Chennamsetty, S. S., Safwan, M., Alex, V., ... & Fernandez, G. (2019). Bach: Grand challenge on breast cancer histology images. Medical image analysis, 56, 122-139.
[33] Bastiaan S. Veeling, Jasper Linmans, Jim Winkens, Taco Cohen and Max Welling. Rotation Equivariant CNNs for Digital Pathology, 2018; arXiv:1806.03962.
[34] Teresa Araújo, Guilherme Aresta, Eduardo Castro, José Rouco, Paulo Aguiar, Catarina Eloy, António Polónia, and Aurélio Campilho, Classification of Breast Cancer Histology Images Using Convolutional Neural Networks, PLOS ONE, 2017. Available at: **https://doi.org/10.1371/journal.pone.0177544**

[35] Atsushi Teramoto, Tetsuya Tsukamoto, Yuka Kiriyama, Hiroshi Fujita, "Automated Classification of Lung Cancer Types from Cytological Images Using Deep Convolutional Neural Networks", BioMed Research International, vol. 2017, Article ID 4067832, 6 pages, 2017. https://doi.org/10.1155/2017/4067832
[36] Kaissis, G.A., Makowski, M.R., Rückert, D. et al. Secure, privacy-preserving and federated machine learning in medical imaging. Nat Mach Intell 2, 305–311 (2020). **https://doi.org/10.1038/s42256-020-0186-1**
[37] Maithra Raghu, Chiyuan Zhang, Jon Kleinberg and Samy Bengio. Transfusion: Understanding Transfer Learning for Medical Imaging, 2019; arXiv:1902.07208.
[38] François Chollet. Xception: Deep Learning with Depthwise Separable Convolutions, 2016; arXiv:1610.02357.
[39] Barret Zoph, Vijay Vasudevan, Jonathon Shlens and Quoc V. Le. Learning Transferable Architectures for Scalable Image Recognition, 2017; arXiv:1707.07012.
[40] Nguyen, Long & Lin, Dongyun & Lin, Zhiping & Cao, Jiuwen. (2018). Deep CNNs for microscopic image classification by exploiting transfer learning and feature concatenation. 1-5. 10.1109/ISCAS.2018.8351550.
[41] Soydaner, Derya. (2020). A Comparison of Optimization Algorithms for Deep Learning. International Journal of Pattern Recognition and Artificial Intelligence. 10.1142/S0218001420520138.

Appendices:

```
lung_model = VGG16(weights = 'imagenet', include_top = False, input_shape = (224,224,3))
num_classes = 3
x = lung_model.output
out1 = GlobalMaxPooling2D()(x)
out2 = GlobalAveragePooling2D()(x)
out3 = Flatten()(x)
x = Concatenate(axis=-1)([out1, out2, out3])
x = Dropout(0.5)(x)
predictions = Dense(num_classes, activation = 'sigmoid')(x)
VGG16_LUNG = Model(lung_model.input, predictions)
VGG16_LUNG.summary()
VGG16_LUNG.compile(optimizer=Adam(0.0001), loss= binary_crossentropy, metrics=['acc'])
h5_path = "VGG_NEW_LUNG.h5"
checkpoint = ModelCheckpoint(h5_path, monitor='val_acc', verbose=1,
                             save_best_only=True, mode='max')
batch_size=64
VGG16_LUNG_HISTORY = VGG16_LUNG.fit(data_gen(lung_train, id_label_map, batch_size),
                        validation_data=data_gen(lung_val, id_label_map, batch_size, augment = False),
                        epochs=10, verbose=1,
                        callbacks=[checkpoint],
                        steps_per_epoch=len(lung_train) // batch_size,
                        validation_steps=len(lung_val) // batch_size)

%matplotlib inline
import matplotlib.pyplot as plt

fig, loss_ax = plt.subplots()

##acc_ax = loss_ax.twinx()

loss_ax.set_ylim([-0.1, 1.1])
###acc_ax.set_ylim([0.0, 1.1])
loss_ax.plot(VGG16_LUNG_HISTORY.history['acc'], 'b', label='train acc')
loss_ax.plot(VGG16_LUNG_HISTORY.history['val_acc'], 'b--', label='val acc')
loss_ax.plot(VGG16_LUNG_HISTORY.history['loss'], 'g', label='train loss')
```

```
preds = []
ids = []
model = VGG16_LUNG
batch_size = 64
test_files = glob('../input/lungfinaldatasets/test_cnc/test_cnc/*.jpeg')
for batch in chunker(test_files, batch_size):
    X = [preprocess_input(cv2.imread(x)) for x in batch]
    ids_batch = [get_id_from_file_path(x) for x in batch]
    X = np.array(X)
    preds_batch = (model.predict(X)).tolist()
    preds += preds_batch
    ids += ids_batch
import pandas as pd
preds = np.array(preds)
VGG16_LUNG_df = pd.DataFrame({'id':ids, 'aca':preds[:,0],'scc':preds[:,1],'n':preds[:,2]})
VGG16_LUNG_df = VGG16_LUNG_df.sort_values(by=['id'])
VGG16_LUNG_df.to_csv("VGG16_LUNG.csv",index = False)
VGG16_LUNG_df.tail()
preds = np.array(VGG16_LUNG_df.iloc[:,1:])
from sklearn.metrics import roc_curve, auc, roc_auc_score, accuracy_score, average_precision_score
labels = ['acc','scc','ben']
#fig, c_ax = plt.subplots(1, 1, figsize=(3, 3))
#for (idx, c_label) in enumerate(labels):
#    fpr, tpr, thresholds = roc_curve(test_Y[:, idx].astype(int), preds[:, idx])
#    c_ax.plot(fpr, tpr, label='%s (AUC:%0.2f)' % (c_label, auc(fpr, tpr)))
#c_ax.legend()
#c_ax.set_xlabel('False Positive Rate')
#c_ax.set_ylabel('True Positive Rate')
#fig.savefig('trained_net.png')
#plt.title('VGG16')

import sklearn
from sklearn.metrics import classification_report
c = np.argmax(preds, axis=1)
#print(classification_report(testYY[:2000],c))
print(classification_report(List,c))

"""
p = VGG16_COLON_HISTORY.history['loss']
q = VGG16_COLON_HISTORY.history['acc']
r = VGG16_COLON_HISTORY.history['val_loss']
s = VGG16_COLON_HISTORY.history['val_acc']
eda = pd.Dataframe({'loss':p, 'acc':q,'val_loss':r,'val_acc':s})
eda.to_csv("eda.csv",index = False)
"""
```

```
lung_model = VGG19(weights = 'imagenet', include_top = False, input_shape = (224,224,3))
num_classes = 3
x = lung_model.output
out1 = GlobalMaxPooling2D()(x)
out2 = GlobalAveragePooling2D()(x)
out3 = Flatten()(x)
x = Concatenate(axis=-1)([out1, out2, out3])
x = Dropout(0.5)(x)
predictions = Dense(num_classes, activation = 'sigmoid')(x)
VGG19_LUNGMAL = Model(lung_model.input, predictions)
VGG19_LUNGMAL.summary()
VGG19_LUNGMAL.compile(optimizer=Adam(0.0001), loss= binary_crossentropy, metrics=['acc'])
h5_path = "VGG19_COLON.h5"
checkpoint = ModelCheckpoint(h5_path, monitor='val_acc', verbose=1,
                             save_best_only=True, mode='max')
batch_size=64
VGG19_COLON_HISTORY = VGG19_LUNGMAL.fit(data_gen(lung_train, id_label_map, batch_size),
                        validation_data=data_gen(lung_val, id_label_map, batch_size, augment = False),
                        epochs=10, verbose=1,
                        callbacks=[checkpoint],
                        steps_per_epoch=len(lung_train) // batch_size,
                        validation_steps=len(lung_val) // batch_size)
%matplotlib inline
import matplotlib.pyplot as plt

fig, loss_ax = plt.subplots()

##acc_ax = loss_ax.twinx()

loss_ax.set_ylim([-0.1, 1.1])
###acc_ax.set_ylim([0.0, 1.1])
loss_ax.plot(VGG19_COLON_HISTORY.history['acc'], 'b', label='train acc')
loss_ax.plot(VGG19_COLON_HISTORY.history['val_acc'], 'b--', label='val acc')
loss_ax.plot(VGG19_COLON_HISTORY.history['loss'], 'g', label='train loss')


loss_ax.plot(VGG19_COLON_HISTORY.history['val_loss'], 'g--', label='val loss')
```

```
preds = []
ids = []
model = VGG19_LUNGMAL
batch_size = 64
test_files = glob('../input/lungfinaldatasets/test_cnc/test_cnc/*.jpeg')
for batch in chunker(test_files, batch_size):
    X = [preprocess_input(cv2.imread(x)) for x in batch]
    ids_batch = [get_id_from_file_path(x) for x in batch]
    X = np.array(X)
    preds_batch = (model.predict(X)).tolist()
    preds += preds_batch
    ids += ids_batch
import pandas as pd
preds = np.array(preds)
VGG16_LUNG_df = pd.DataFrame({'id':ids, 'aca':preds[:,0],'scc':preds[:,1],'n':preds[:,2]})
VGG16_LUNG_df = VGG16_LUNG_df.sort_values(by=['id'])
VGG16_LUNG_df.to_csv("VGG16_LUNG.csv",index = False)
VGG16_LUNG_df.tail()
preds = np.array(VGG16_LUNG_df.iloc[:,1:])
from sklearn.metrics import roc_curve, auc, roc_auc_score, accuracy_score, average_precision_score
labels = ['acc','scc','ben']
fig, c_ax = plt.subplots(1, 1, figsize=(3, 3))
for (idx, c_label) in enumerate(labels):
    fpr, tpr, thresholds = roc_curve(test_Y[:, idx].astype(int), preds[:, idx])
    c_ax.plot(fpr, tpr, label='%s (AUC:%0.2f)' % (c_label, auc(fpr, tpr)))
c_ax.legend()
c_ax.set_xlabel('False Positive Rate')
c_ax.set_ylabel('True Positive Rate')
fig.savefig('trained_net.png')
plt.title('VGG19')
import sklearn
from sklearn.metrics import classification_report
c = np.argmax(preds, axis=1)
print(classification_report(testYY,c))
```

```
lung_model = Xception(weights = 'imagenet', include_top = False, input_shape = (224,224,3))
num_classes = 3
x = lung_model.output
out1 = GlobalMaxPooling2D()(x)
out2 = GlobalAveragePooling2D()(x)
out3 = Flatten()(x)
x = Concatenate(axis=-1)([out1, out2, out3])
x = Dropout(0.5)(x)
predictions = Dense(num_classes, activation = 'sigmoid')(x)
Xception_LUNG = Model(lung_model.input, predictions)
Xception_LUNG.summary()
Xception_LUNG.compile(optimizer=Adam(0.0001), loss= binary_crossentropy, metrics=['acc'])
h5_path = "Xception_LUNG.h5"
checkpoint = ModelCheckpoint(h5_path, monitor='val_acc', verbose=1,
                             save_best_only=True, mode='max')
batch_size=64
Xception_LUNG_HISTORY = Xception_LUNG.fit(data_gen(lung_train, id_label_map, batch_size),
                        validation_data=data_gen(lung_val, id_label_map, batch_size, augment = False),
                        epochs=10, verbose=1,
                        callbacks=[checkpoint],
                        steps_per_epoch=len(lung_train) // batch_size,
                        validation_steps=len(lung_val) // batch_size)
%matplotlib inline
import matplotlib.pyplot as plt

fig, loss_ax = plt.subplots()

##acc_ax = loss_ax.twinx()

loss_ax.set_ylim([-0.1, 1.1])
###acc_ax.set_ylim([0.0, 1.1])
loss_ax.plot(Xception_LUNG_HISTORY.history['acc'], 'b', label='train acc')
loss_ax.plot(Xception_LUNG_HISTORY.history['val_acc'], 'b--', label='val acc')
loss_ax.plot(Xception_LUNG_HISTORY.history['loss'], 'g', label='train loss')


loss_ax.plot(Xception_LUNG_HISTORY.history['val_loss'], 'g--', label='val loss')



loss_ax.set_xlabel('Epoch')
loss_ax.set_ylabel('loss / accuracy')
### acc_ax.set_ylabel('accuray')

loss_ax.legend(loc='center right')


plt.title("Xception")
plt.show()
```

```
preds = []
ids = []
model = Xception_LUNG
batch_size = 64
test_files = glob('../input/lungfinaldatasets/test_cnc/test_cnc/*.jpeg')
for batch in chunker(test_files, batch_size):
    X = [preprocess_input(cv2.imread(x)) for x in batch]
    ids_batch = [get_id_from_file_path(x) for x in batch]
    X = np.array(X)
    preds_batch = (model.predict(X)).tolist()
    preds += preds_batch
    ids += ids_batch
import pandas as pd
preds = np.array(preds)
VGG16_LUNG_df = pd.DataFrame({'id':ids, 'aca':preds[:,0],'scc':preds[:,1],'n':preds[:,2]})
VGG16_LUNG_df = VGG16_LUNG_df.sort_values(by=['id'])
VGG16_LUNG_df.to_csv("VGG16_LUNG.csv",index = False)
VGG16_LUNG_df.tail()
preds = np.array(VGG16_LUNG_df.iloc[:,1:])
from sklearn.metrics import roc_curve, auc, roc_auc_score, accuracy_score, average_precision_score
labels = ['acc','scc','ben']
fig, c_ax = plt.subplots(1, 1, figsize=(3, 3))
for (idx, c_label) in enumerate(labels):
    fpr, tpr, thresholds = roc_curve(test_Y[:, idx].astype(int), preds[:, idx])
    c_ax.plot(fpr, tpr, label='%s (AUC:%0.2f)' % (c_label, auc(fpr, tpr)))
c_ax.legend()
c_ax.set_xlabel('False Positive Rate')
c_ax.set_ylabel('True Positive Rate')
fig.savefig('trained_net.png')
plt.title('Xception')
import sklearn
from sklearn.metrics import classification_report
c = np.argmax(preds, axis=1)
print(classification_report(testYY,c))
```

```
lung_model = NASNetMobile(weights = 'imagenet', include_top = False, input_shape = (224,224,3))
num_classes = 3
x = lung_model.output
out1 = GlobalMaxPooling2D()(x)
out2 = GlobalAveragePooling2D()(x)
out3 = Flatten()(x)
x = Concatenate(axis=-1)([out1, out2, out3])
x = Dropout(0.5)(x)
predictions = Dense(num_classes, activation = 'sigmoid')(x)
NASNetMobile_LUNG = Model(lung_model.input, predictions)
NASNetMobile_LUNG.summary()
NASNetMobile_LUNG.compile(optimizer=Adam(0.0001), loss= binary_crossentropy, metrics=['acc'])
h5_path = "NASNetMobile_LUNG_last.h5"
checkpoint = ModelCheckpoint(h5_path, monitor='val_acc', verbose=1,
                             save_best_only=True, mode='max')
batch_size=64
NASNetMobile_LUNG_HISTORY = NASNetMobile_LUNG.fit(data_gen(lung_train, id_label_map, batch_size),
                        validation_data=data_gen(lung_val, id_label_map, batch_size, augment = False),
                        epochs=10, verbose=1,
                        callbacks=[checkpoint],
                        steps_per_epoch=len(lung_train) // batch_size,
                        validation_steps=len(lung_val) // batch_size)
%matplotlib inline
import matplotlib.pyplot as plt

fig, loss_ax = plt.subplots()

##acc_ax = loss_ax.twinx()

loss_ax.set_ylim([-0.1, 1.1])
###acc_ax.set_ylim([0.0, 1.1])
loss_ax.plot(NASNetMobile_LUNG_HISTORY.history['acc'], 'b', label='train acc')
loss_ax.plot(NASNetMobile_LUNG_HISTORY.history['val_acc'], 'b--', label='val acc')
loss_ax.plot(NASNetMobile_LUNG_HISTORY.history['loss'], 'g', label='train loss')


loss_ax.plot(NASNetMobile_LUNG_HISTORY.history['val_loss'], 'g--', label='val loss')



loss_ax.set_xlabel('Epoch')
loss_ax.set_ylabel('loss / accuracy')
### acc_ax.set_ylabel('accuray')

loss_ax.legend(loc='center right')


plt.title("NASNetMobile")
plt.show()
```

```
preds = []
ids = []
model = NASNetMobile_LUNG
batch_size = 64
test_files = glob('../input/lungfinaldatasets/test_cnc/test_cnc/*.jpeg')
for batch in chunker(test_files, batch_size):
    X = [preprocess_input(cv2.imread(x)) for x in batch]
    ids_batch = [get_id_from_file_path(x) for x in batch]
    X = np.array(X)
    preds_batch = (model.predict(X)).tolist()
    preds += preds_batch
    ids += ids_batch
import pandas as pd
preds = np.array(preds)
VGG16_LUNG_df = pd.DataFrame({'id':ids, 'aca':preds[:,0],'scc':preds[:,1],'n':preds[:,2]})
VGG16_LUNG_df = VGG16_LUNG_df.sort_values(by=['id'])
VGG16_LUNG_df.to_csv("VGG16_LUNG.csv",index = False)
VGG16_LUNG_df.tail()
preds = np.array(VGG16_LUNG_df.iloc[:,1:])
from sklearn.metrics import roc_curve, auc, roc_auc_score, accuracy_score, average_precision_score
labels = ['acc','scc','ben']
fig, c_ax = plt.subplots(1, 1, figsize=(3, 3))
for (idx, c_label) in enumerate(labels):
    fpr, tpr, thresholds = roc_curve(test_Y[:, idx].astype(int), preds[:, idx])
    c_ax.plot(fpr, tpr, label='%s (AUC:%0.2f)' % (c_label, auc(fpr, tpr)))
c_ax.legend()
c_ax.set_xlabel('False Positive Rate')
c_ax.set_ylabel('True Positive Rate')
fig.savefig('trained_net.png')
plt.title('NASNetMobile')
import sklearn
from sklearn.metrics import classification_report
c = np.argmax(preds, axis=1)
print(classification_report(testYY,c))
```

```
lung_train, lung_val = train_test_split(labeled_files, test_size=0.2, random_state=101010)
def data_gen(list_files, id_label_map, batch_size, augment=True):
    seq = get_seq()
    while True:
        shuffle(list_files)
        for batch in chunker(list_files, batch_size):
            X = [cv2.imread(x) for x in batch]

            Y = [id_label_map[get_id_from_file_path(x)] for x in batch]
            if augment:
                X = seq.augment_images(X)
            X = [preprocess_input(x) for x in X]


            yield np.array(X), np.array(Y)
```

```
ids_lungaca=[]
for i in range(1,1001):
    y = "lungaca" + str(i) + ".jpeg"
    ids_lungaca.append(y)
ids_lungn=[]
for i in range(1,1001):
    y = "lungn" + str(i) + ".jpeg"
    ids_lungn.append(y)
ids_lungscc = []
for i in range(1,1001):
    y = "lungscc" + str(i) + ".jpeg"
    ids_lungscc.append(y)
labels_aca = []
for i in range(0,1000):
    labels_aca.append(0)
labels_n = []
for i in range(0,1000):
    labels_n.append(2)
labels_scc = []
for i in range(0,1000):
    labels_scc.append(1)
idss = ids_lungaca + ids_lungn + ids_lungscc
len(idss)
labels = labels_aca + labels_n + labels_scc
len(labels)
import pandas as pd
test_y = pd.DataFrame({'id':idss, 'label':labels})
print(test_y.head())
test_y.tail()
#lung_mal.to_csv("lung_mal.csv",index = False)
```

Figure 16: Input Codes of Inputs and Prediction of CNN models