CCE 3050

Computer Aided Diagnosis of Breast Cancer using Convolutional Neural Networks

Student Name: Mohammed Taha

Student Number: M00608647

Supervisor: Mohammed Osman

Module Leader: Xiaochun Cheng

A thesis submitted in partial fulfilment of the requirements for the degree of Bachelor of Science in

BEng Computer Systems Engineering Hons

Front Cover Sheet & Statement of Originality of Work

Faculty of Science and Technology

I am Mohammed Taha

Student ID Number: M00608647

Supervisor name: Mohammed Osman

Module Number: CCE3050

Module Title: Individual Project

Module Leader: Xiaochun Cheng

I hereby confirm that the work presented here in this report and in all other associated material is wholly my own work. I confirm that the report has been submitted to TURNITIN and that the TURNITIN results are on CD/DVD attached to this report. I agree to assessment for plagiarism.

Signature: Mohammed Taha

Date: 26th April, 2019

Project Title: Computer Aided Diagnosis of Breast Cancer using Convolutional Neural Networks

Table of Contents

Abstract

Body temperature is a lesser known but very useful parameter for disease diagnosis according to many researchers and medical professionals. In early medicine, Physicians would diagnose patients with wet mud or slurry clay. When either of these was spread over the body, the part that would dry up first was considered the diseased part. In the modern day, this can be achieved with thermographic cameras that produce images of heat signatures. This is done by detecting the long infrared range of the electromagnetic spectrum. Areas of inflammation and lymphatic congestion which are precursors to cancer can be identified with thermography without any radiation or contact. This can be used as a great advantage for screening of patients before any major symptoms have surfaced. On the technological aspect of this project, Machine learning is the field of computer science that uses statistical techniques to give computer systems the ability to learn with data, without being explicitly programmed. Machine Learning can aid to this cause by reading these thermal scans and recognizing areas of suspicion that need to be further inspected by a doctor. Thermal imaging being a comparatively cheaper alternative to other tests that require expensive equipment and machinery can provide a quicker and harmless procedure for clinics and hospitals to implement

Chapter 1

Introduction

1.1 Introduction

This chapter brings to light the essence of this project. It introduces the background and motivation behind the topic of research which is automation of breast cancer detection using thermal infrared imaging and convolutional neural networks. The problem definition, Aim, Scope of work and research methodology are also defined in this chapter to fully understand the thought process and expected outcomes of this project.

1.2 Background

As humans progress in all aspects of life, Disease still prevails. While there are continuous improvements to diagnostic methods and medicine, these are more often quite expensive and cause a significant impact to the lives of average-income and low-income families. Even after these treatment procedures, the guarantee of being cured is not ensured. Breast cancer is one of these diseases that continue to plague women in every part of the world. It is caused when the breast cells begin to grow abnormally and very rapidly. These abnormal cells eventually accumulate and form a malignant tumor which can be felt as a lump (Mayo Clinic, 2019). According to World Health Organization, an estimated number of 10 lakh cases of breast cancer are reported in India every year. Out of these 10 lakh cases reported, 5 lakh patients passed away. WHO has predicted that this number may increase up to 5 times more by 2025 with an increase of 23% in women and 19% in men (Mukerji, 2019). Currently the most popular way of diagnosing for breast cancer is a mammogram. Mammography is a diagnostic technique that utilizes low dose x-rays to detect malignant tumors. The procedure itself is performed with the assistance of a radiologist and an x-ray machine which requires the patients breasts to placed on the machine with a transparent plastic layer between the patient and the machine. Women age 45 onwards are advised to getting screened every 1 to 2 years as a preventive measure against breast cancer. Unfortunately, the number of false positives that occur in mammograms performed each year for 10 years is 49.1% (Kandlikar, 2019). Breast cancer is classified into 4 stages, Stage 0 being non-invasive, Stage 1 and 2 being early stages and Stage 4 and 5 is considered as the late stage. If the tumor is detected in the early stages, it can reduce the cost of treatment drastically. Despite the recommended testing frequency for breast cancer, the cost of such screening tests in underprivileged countries restricts women to get tested regularly. In India, a regular mammogram can amount up to Rs. 2000 and a digital mammogram can amount up to Rs. 8000 (Mukerji, 2019). Besides this, the invasive nature of these tests is also a contributing factor to the reason mammograms are avoided until prescribed by a doctor. During the early days of medicine, physicians used slurry clay or wet mud as a diagnosis tool. After the clay is applied, the region that dried up the quickest was considered as the diseased part. Temperature as an indicator is a lesser known but very useful parameter for detection of possible diseases. With current technology, a much more sophisticated result can be achieved with the help of a thermal scanner. These scanners produce images of heat signatures by detecting the long infrared range of the electromagnetic spectrum. Certain precursors to cancer such as lymphatic congestions and areas of inflammation can be detected significantly earlier than traditional signs without the need for any contact or radiation. Using thermal can prove to be a great asset to detect areas of concern before harmful symptoms begin to show.

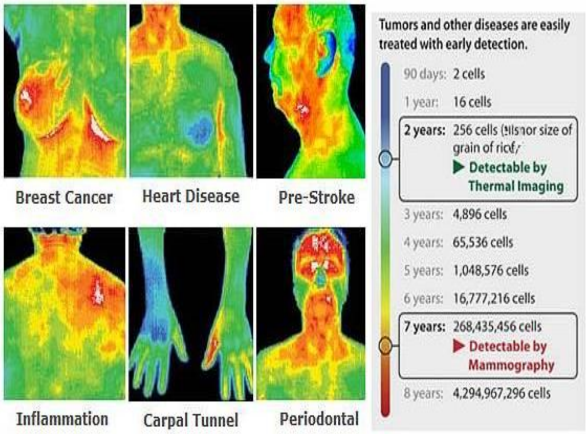

Breast Cancer
Heart Disease
Pre-Stroke
Inflammation
Carpal Tunnel
Periodontal
Tumors and other diseases are easily treated with early detection.
90 days: 2 cells
1 year: 16 cells
2 years: 256 cells (size of grain of rice)
Detectable by Thermal Imaging
3 years: 4,896 cells
4 years: 65,536 cells
5 years: 1,048,576 cells
6 years: 16,777,216 cells
7 years: 268,435,456 cells
Detectable by Mammography
8 years: 4,294,967,296 cells

Figure 1 (O2 Wellness, 2019)

As capturing thermal images only requires an infrared camera which costs considerably less when compared to the equipment required for a mammogram. On the other hand, the field of medical diagnosis is improving rapidly with the integration of deep learning. Deep learning is a division of machine learning where algorithms are designed based on the working of the human brain. This part of the brain that is mathematically replicated is known as neural networks. Over the last few years, artificial neural networks have been applied to many research projects involving medical diagnosis. The result of these research papers has observed results that are equivalent to and sometimes even outperform doctors (Towards Data Science, 2019).

1.3 Problem Definition

Early detection of symptoms can reduce cost of treatment tremendously. Yet, majority of the general public do not take precautions until more serious symptoms begin to show. People refrain from taking screening tests due to the cost and complexity of it, fear of pain and/or discomfort during the test and the time-consuming nature of majority of tests. Furthermore, rural areas are considerably unaware as compared to developed cities. Awareness campaigns may not have resources required to conduct mass screening on a regular basis without having to spend a fortune. The Medibulletin states in an article about Indian villages “Villagers do not have access to statistics but say at any given point there are 200-250 cancer patients among the 1700 families in the village”. With such high numbers, it is evident that a faster and cheaper screening method would save countless lives (Health news, Medibulletin, 2019).

The rates of survival for breast cancer are very different worldwide, they can range from 80% or above in Sweden, Japan and North America to an estimated 60% in average income countries and below 40% in developing and low income countries. The low survival rates in these low income countries is correlated to the the lack of early detection programs and a lack of adequate diagnosis and treatment facilities which result in a large portion of women being diagnosed with breast cancer in the late-stage, reducing the survival rate (World Health Organization, 2019).

1.4 Aim

The main aim of this project is to provide a quicker, cheaper and autonomous alternative to tedious and expensive screening methods for diagnostic purposes in under privileged countries as well as the rest of the world. This goal is to be carried out using the combination of Thermal infrared images and deep learning. This should allow for mass awareness campaigns with a quicker and non-invasive screening technique while restricting to a low budget in terms of resources and staff.

1.5 Objectives

- To acquire dataset consisting of infrared images.
- To segregate the dataset into healthy and sick patients.
- To sort and clean the acquired dataset into training, testing and validation set.
- To train and test the convolutional neural network.
- To validate the results by passing them through the trained model.
- To plot a confusion matrix to compare the true positives, false positive, true negative and false negatives.

1.6 Scope of Work

The work done in this project has the potential to impact lives globally but more specifically in low income countries that require access to cheaper methods of screening to improve survival rates.

The occurrence of Breast cancer is widely observed in women of both developed and less developed countries. There is an approximation that all over the world, there are over 508 000 women that have died in 2011 alone due to breast cancer. While breast cancer is assumed to be a disease that only occurs in the developed world, the alarming truth is that roughly 50% of cases regarding breast cancer and 58% of deaths are arise from under developed and low income countries (World Health Organization, 2019).

1.7 Research Methodology

The research done in this project revolves deeply around data collection and data analysis. The researcher collects a dataset with a large number of infrared images which can later be used to classify whether new patients are affected by breast cancer based on the infrared image provided.

As the dataset collected is the primary focus of this research, the research methodology being used is partially Quantitative. Quantitative research is research that involves using of Statistical, Mathematical and most importantly, computational power to analyze the large amount of data collected. The paper is classified as partially quantitative as it relies on collection of data as its input and produces an output which will allow the researcher to understand how significant the features of a tumor are in a patient with breast cancer and if these features form a repetitive pattern which can be detected by a convolutional neural network.

The areas of research covered in this paper are in the areas of radiology and more specifically thermal infrared imaging and its benefits over the more mainstream diagnostic methods which are used currently for breast cancer detection. The data collected can be analyzed accurately only after understanding how thermography is able to detect tumors in patients. Extensive research is required in this field to better understand how to proceed.

Artificial Neural networks are an area of machine learning which was developed based on inspiration from how the human brain classifies objects. The classification is executed via algorithms that allow the machine to learn by incorporating new data. These classification models depend on the patterns that can be recognized due to their repeated occurrence. To meet the specific requirements of the research in this project, many classification models will be researched to select the model best suited for this task.

Chapter 2

Literature Review

2.1 Introduction

The following section digs deeper into the research done related to the project topic. It breaks down the relationship between the research done in the respective fields and how the combination of the work is relevant to the topic.

The literature reviewed for this project can be divided into 3 topics:

Topic 1	Healthcare in Rural communities
Topic 2	Thermography
Topic 3	Automation in Healthcare

Table 1

2.2 Topic 1: Healthcare in Rural communities

Author	Title	Year	Type
Gupta, s. et al	Big Data Lakes Can Support Better Population Health for Rural India - Swastha Bharat	2016	Research Paper
Gupta, a. et al	A review of breast cancer awareness among women in India: Cancer literate or awareness deficit?	2015	Journal

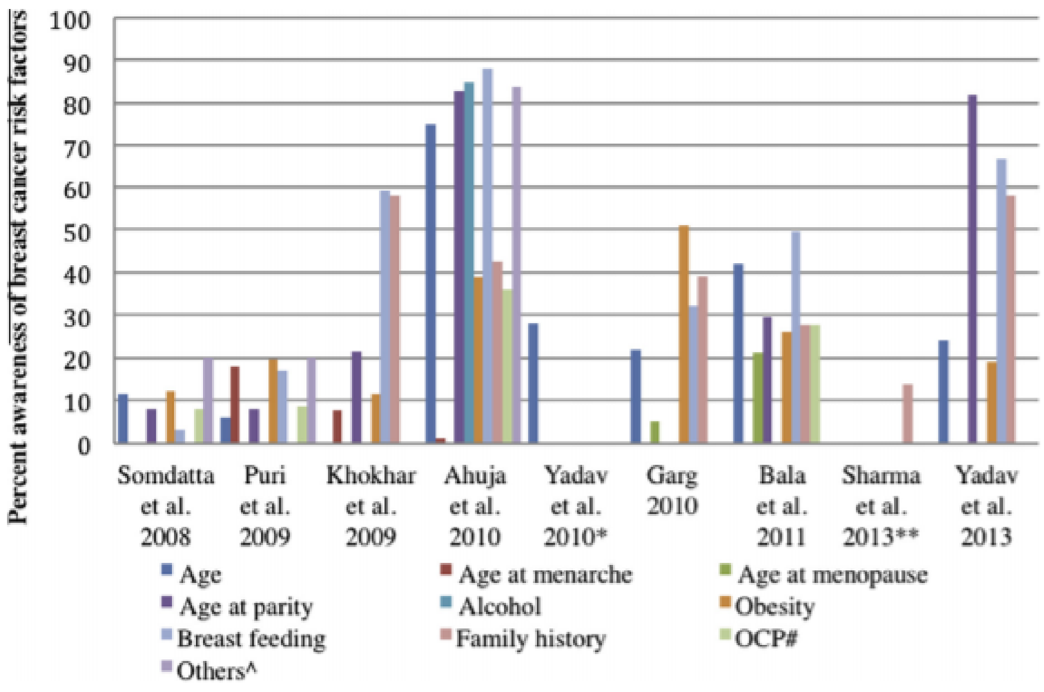
Table 2

- **Big Data Lakes Can Support Better Population Health for Rural India - Swastha Bharat**

The research paper published in 2016 by research scholars in Mewar University, India discusses in depth how communities living metropolitan cities in India have access to top quality health care facilities, the folk in rural and remote districts of India often have very poor access to Healthcare. This lack of exposure to proper healthcare is a growing problem with the rise of population in Asia. While India is one of the growing economies of today, 29.5% of the population is below poverty line. Part of the reason for this inadequate access to healthcare is the expensive and time-consuming aspect of preventive diagnosis. This has made it a necessity to come up with unique and innovate solutions to solve an ever-increasing problem. The paper aims to bring to light how Big Data can make the health care industry in India more economical, Cost effective and hence reach out the larger mass of the population. As data is always being produced, patient data is stored and managed electronically in a dataset known as the Health Data Set. According to the authors, the performance of the health industry can improve itself by Analyzing the stored data to boost treatment quality, Managing the revenue prices by reducing unnecessary tests and improve preventive care and increase patient satisfaction.

This paper perfectly describes the foundation of the problem which is poor access to healthcare and preventive diagnosis in the more rural communities of India and likewise other countries as well. While the paper does describe the problem sufficiently, it only describes the working of Big Data Analytics. No concrete solution, testing or result were produced in the paper.

- **A review of breast cancer awareness among women in India: Cancer literate or awareness deficit?**
  The content of this journal publication is a deeper look into the awareness of breast cancer and cancer literacy in India where majority of the population is classified as lower middle income or low income. The authors of this journal conducted their research by searching and analyzing data from several databases and studies which range from across the lowest income areas to the more well settled areas of India. This data consisted of information of 7000 women ranging from age 15-70. The women selected for the sample were from all walks of life in India, ranging from low income regions to the more well settled regions of India. The authors discover low levels of awareness amongst most women from this sample regardless of their socio-economic and educational backgrounds. A graph of the information collected over the last couple of years provides an understanding that the level of literacy for early cancer detection and preventive measures.



A. Gupta et al. / European Journal of Cancer 51 (2015) 2058–2066

Figure 2

According to the graph, the various studies conducted over the years were plotted in chronological order to study the awareness of causes of breast cancer over the years. The study found no relevant increase in cancer literacy. A low spike of awareness was noticed during a small period of time which did not reflect over time.

Due to this lack of awareness, the recommended frequency of testing is not met which results in late stage detection of breast cancer. The women of India need to be more aware of the risk facts for breast cancer and practice habits for scheduled testing for prevention. This needs to be put into effect by official authorities in the form of awareness campaigns and mass screening tests encouraging women to get tested.]

2.3 Topic 2: Thermography

Author	Title	Year	Type
Frize, M. et al	Processing Thermal Images to detect breast cancer and assess pain	2003	Research Paper
Chekmenev et al	Non-contact, Wavelet-based Measurement of Vital Signs using Thermal Imaging	2005	Research Paper
Sruthi, S. et al	A low cost thermal imaging system for medical diagnostic applications	2015	Research Paper

Table 3

- **Processing Thermal Images to detect breast cancer and assess pain**
  Thermography was used in 1960 to detect breast cancer and 1980 to assess pain. Doctors discovered that there was a significant difference in the temperature distribution in patients with cancer symptoms as opposed to healthy patients. The current popular test for breast cancer diagnosis was a mammogram which required the use of ionizing radiation and had an uncomfortable approach. Early researchers such as Gautherie et al believed that thermography had the potential to detect early changes in the blood flow which indicated asymmetrical breasts. Another disadvantage of mammography was it was unable to detect symptoms in women 25 to 40. Research proved the sensitivity and results of thermography were the same as mammography. Analyzing these thermal images by identifying patterns was a Difficult and time-consuming task to be done by the human eye. To automate this process, Standard deviation was used. Data was collected at the Moncton Hospital between march and November 1984 with a first-generation thermographic camera. The process of comparing was the Entire Left and right breast were compared and then the breast was divided into four quadrants and the temperature of each of these quadrants was compared. After comparing regions and being graded by

the doctors, The Results were split into three categories namely, Benign, Cancerous and Fibrocystic. The paper was alert about the mainstream use of mammography and proved with research that Medical thermography can provide an equally compelling result. This can be further proved with clinical trials. The information in this paper shines light of the use of Thermography for the purposes of detecting breast cancer and cancerous symptoms in patient. The use of mathematics was also proven using standard deviation. While this is an older technique of automation, it is the base and foundation of modern day Machine learning. The work in this research paper is directly related to this final project. Future work would include enhancing technique to achieve a much faster and noninvasive method for breast cancer detection

- **Non-contact, Wavelet-based Measurement of Vital Signs using Thermal Imaging**
  This research paper evaluates the potential of using non-intrusive remote passive thermal imaging for measurements of human vital signs. Breathing causes a significant change in temperature in the nasal area, which periodically appears in thermal images. This is combined with wavelet analysis to extract the pulse and respiration. This is done by taking thermal images of the neck and face area. These images are taken in different scales and the scale that carries the most relevant information in chosen. The results show a non-contact wavelet-based frame work to measure human vital signs. The result to this research provides a 100% accuracy for both breathing and heart rate.
  This research successfully used thermography to detect human vital signs. Disadvantages include additional noise and patterns that are not relevant to vital sign measurement, but the results are successful nevertheless. All the testing and finding were carried out by the author themselves and is first hand. While the research in this paper is not completely relevant to this project, it is a step in the direction that Medical Thermography is an efficient and working technique for noninvasive screening.

- **A low cost thermal imaging system for medical diagnostic applications**
  This paper presents a low cost thermal imaging camera system designed specifically for medical diagnostic applications. An array detects the infrared radiations from the human body which are converted to electronic signals. These signals are further converted to matrix which represents temperature value. The set up includes a thermophile-based sensor and a microcontroller. The aim of this project is to reduce the cost required to create thermal images. On average, Industrial thermal cameras cost an average of 5000$ while this entire setup can be produced in an average of 150$.
  The author successfully created the device aimed to create. The goals are relevant to this final project as the ultimate target is to be able to reduce the cost required to conduct screenings in low income communities. While the final product is efficient, it may not be able to capture the high definition thermal images required by medical professionals. The project is a step in the direction of bringing thermography to the mainstream.

2.4 Topic 3: Automation in Medicine

Author	Title	Year	Type
Roychowdhury, S. et al	DREAM: Diabetic Retinopathy Analysis Using Machine Learning	2014	Research Paper

Table 4

- **DREAM: Diabetic Retinopathy Analysis Using Machine Learning**
  Summary: DREAM, Diabetic Retinopathy Analysis Using Machine Learning (2014) presented a screening system aided by Machine Learning. The contents of this paper describe the successful creation and testing of a 3 stage automated system that aims classify retinopathy lesions from non-lesions and hence create a model that generates a severity grading for diabetic retinopathy. As very thoroughly explained in this paper, Diabetic retinopathy is a serious disorder which could potentially lead to vision loss if not intervened early. DR(Diabetic Retinopathy) is caused by the blood vessels being damaged in the retina due to diabetes. Despite 4.4 million Americans suffering from DR, almost 50% do not undergo any form of screening tests and a major cause for that is insufficient referrals, economic hinderances and lack of access to eye care. The main contribution this research has provided is reducing the number of features needed to identify retinopathic legions and hence produce a result that states how severe the diabetic retinopathy is. The methods used to achieve this are powerful algorithms such as Gaussian Mixture Model, K-nearest neighbor, Support vector machine, AdaBoost and a combination of these methods. Gaussian Mixture Method and K-nearest neighbor were found to produce the most accurate results for bright and red lesion classification. Two Diabetic retinopathy datasets were used, namely DIARETDB1 which contains 89 images out of which 28 training and 61 test images. The second dataset known as MESSIDOR contains 1200 images which have each been manually examined and graded for severity. The combination of these datasets provides a good mixture of images to provide an in-depth variety of images that result in a precise model. The 3 stages used in this automated system starts with the first stage where a minimumintensity maximum-solidity algorithm is invoked to detect regions in the retina that consist of red legions and bright legions, the two main indicators of DR. Stage 2 is broken down into 2 steps, the first being classifications into true lesions and non-legions followed by step 2 where the bright regions and red regions are classified further into two types each. In stage 3, the number of red legions and bright legions are counted and combined using a combination function to generate a severity grade. The algorithms used for this classification results in a model that outperforms all existing models. Compared to the previous record, which was 96% sensitivity, 51% specificity and 0.875 AUC (Area Under ROC Curve) This model yields a

result of 100% sensitivity, 53.16% specificity and an AUC of 0.904. Further, the number of images rejected has been brought down to 0% as compared to previous models such as the VA Screening System which rejected 18% of images and the Meadalytix System which reject 5% of images. One of the ways the DREAM model has achieved this is by increasing image sharpness and contrast to obtain a clear picture. Upon testing on the field, A Hospital in Scotland have reported diagnosis in 6 seconds and reduction of workload by 25%. The work done in this research is clearly thorough and utilizes existing datasets and Machine Learning technology to create accurate and precise results which are currently being used in hospitals worldwide. The combination of Machine learning with medical scans allows to speed up the process of diagnosis and allows for more exposure to the common public using cheaper and more efficient methods. This bridges a major gap between complicated tests and lack of access. The work conducted in this research is directly relevant as it used the same techniques and had the goal to automate a process to create more exposure.

Chapter 3

System Analysis and Design

3.1 Introduction

The content of this chapter includes a thorough description of the data, tools, dependencies and most importantly, classification models required to be able to achieve conclusive results in this research project. Besides the requirements, the chapter also describes how the pipeline for the project is theorized. It provides the reader with an understanding of the steps intended to produce a result.

3.2 Requirement Specification

3.2.1 Jupyter Notebooks

The project is a combination of a medical diagnostic technique that is not practiced very widely and a category of machine learning that allows the researcher to produce a classification model that can differentiate between healthy and sick patients. To implement this project, a workspace suited for the specific nature of this task is required. The most popular language used for machine learning projects is python. For this project, python programming language will be used along with a scientific notebook known as Jupyter Notebook. Jupyter notebooks which was previously termed iPython notebooks, is the most common tool used by data scientists all over the world. It is an extremely powerful, versatile and easy to use workspace which can be shared in the form of a simple file. It's an open source web application that where the code written can be run step by step including visualizations which allows users to be able to look at the outcome produced without having to leave the environment.

Jupyter notebooks were introduced in 2014 as a part of a project known as project Jupyter. It was announced by the creator Fernando Perez, who is a physicist and software developer who originally created IPython. He announced Jupyter labs as the next phase of the IPython project. The most notable feature being Jupyter labs are language agnostic and its name itself is comprised of the three main programming languages intended to be used on it, Julia, Python and R. Besides these three languages, the workspace supports several other languages as well. Soon after the announcement, the support of .ipynb files was added to GitHub as well (Perez, 2019). The reason for its popularity amongst the data science community is due to the nature of the work. Data science work requires lots of analysis and visual outputs to be investigated before moving on to the statistical modelling. Jupyter contains a very special feature that allows users to write code in specific cells and view the output instantly below the cell if necessary. This dynamic nature of work really comes in handy for data scientists while performing end to end tasks such as cleaning of the data, building and training the models as the exact cell which is causing an error can be inspected. Jupyter notebooks can be installed separately or along with an entire distribution designed specifically for scientific computing consisting of open source software known as Anaconda distribution.

3.2.2 Python Libraries

Python is the most popular language for machine learning projects. Working with an image dataset can be a complicated task but can be executed with a single line of code in python. Python is an extremely easy language to use which can be helpful while executing complex topics. It contains a package for every possible task. Packages are namespaces that can be imported. These namespaces contain multiple modules within itself. When these packages are imported, functions already defined inside the package can be used in the project without the hassle of having to program them yourself. The main libraries used for this project are numpy, pandas, sklearn, itertools, matplotlib, opencv and keras.

Numpy is a package used in every in every single data science project. It is one of the most fundamental packages used. Numpy is used for scientific computing and can be used for Multidimensional arrays, Fourier transformations, shape manipulation, linear algebra and is an alternative to MATLAB in python (Numpy.org, 2019).

Pandas, which stands for python data analysis library is an open source library that is used for data structures and data analysis. It has been described as the perfect tool to bridge the gap between ad hoc analysis and production quality code. Similar to numpy, pandas is also a library that is used for almost every data science project (Pandas.pydata.org, 2019).

Matplotlib is a library in python that allows the user to plot 2D charts and graph. This is an essential tool as visualizations are an important aspect of data science. plots, histograms, power spectra, confusion matrices, bar charts, scatterplots, etc. can be generated by simply typing a few lines of code (Matplotlib.org, 2019).

OpenCV which stands for Open Source Computer Vision is a library that that was made to provide a common infrastructure to assist with computer vision projects and to help accelerate the use of computer understanding for commercial products. It contains more than 2500 algorithms that leverage classic techniques as well as modern day computer vision techniques. These algorithms can detect a wide variety of objects and is very effective. It is currently being by all major companies working with computer vision (Opencv.org, 2019).

Keras is a high level neural network API which works on top of backend processes such as tensor flow, CNTK and Theano. It was developed for the specific reason to be able to enable fast experimentation on computers. With the integration of Keras, data scientists are able to prototype fast and easily. Its used in this project due to its support for convolutional neural networks and its smooth operation on both CPU and GPU (Keras.io, 2019).

3.2.3 Convolutional neural network

Artificial Neural Networks is a division of machine learning which has been developing and improving over the years. Researchers have found that by leveraging the availability of big data, these neural networks can predict outputs and identify patterns with outstanding precision, accuracy and speed. A neural network is a collection of neurons or nodes that are connected to each other as seen in fig 3. These neurons can pass signals from one unit to the other and can also be tuned. the working of a neural network is replicated from the structure and functioning of the human brain which allows them to make accurate predictions very similar to humans and also learns over repeated iterations.

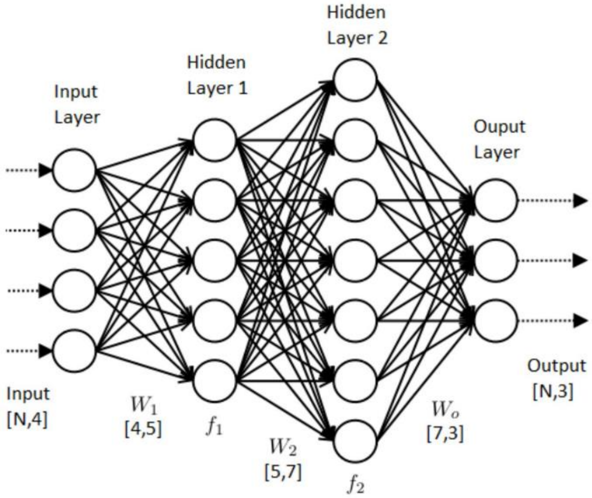


Figure 3 (Ahire, 2019)

On the other hand, Convolutional Neural Networks are used for classification and identification of images. Convolutional Neural Networks were formed as a result of bio-mimicry. Researchers, D. H. Hubel and T. N. Wiesel suggested a system by which mammals distinguish between the many objects around them. They proposed that they used a layer of neurons in the brain. This ideology inspired engineers to form algorithms that would be able to replicate the communication of neurons for classification of objects. The assumption was that within the virtual cortex of these mammals, there existed “simple cells”. These cells generated simplistic responses which in turn were constructed into complex functional responses which were developed by “complex cells”. This natural phenomenon is recreated in the form of a Deep learning algorithm. They require an input image and learnable weights to be able to differentiate between images. It goes through a pipeline of 6 essential functions which output a result based on its training. (Raval, 2019)

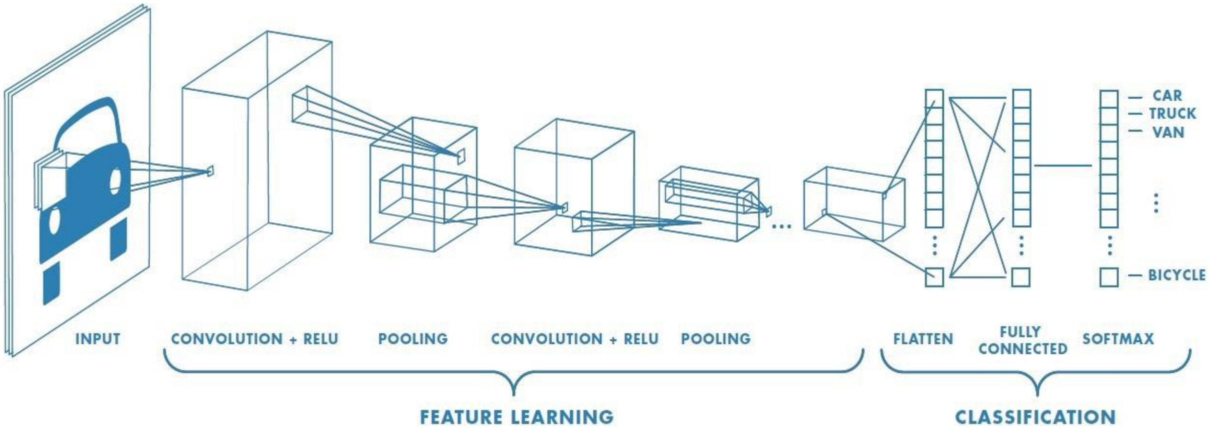
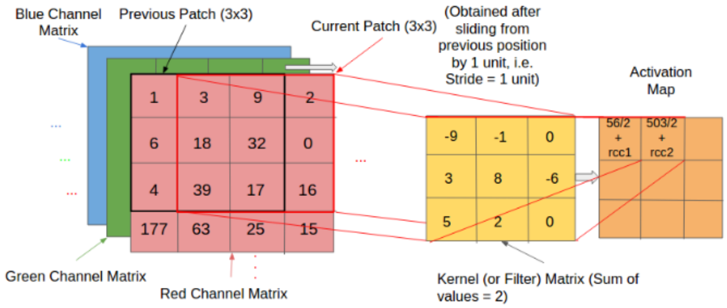


Figure 4 (Saha, 2019)

When an image enters the pipeline, it comes with certain RGB values. These RBG images are separated by three planes which are namely, Red, Green and Blue. Besides RGB, an image can also exist in HSV, CMYK and Grayscale. Due to the color values and added complexity of the image, simple matrix multiplication to flatter the image will not suffice. The convolutional network is able to comprehend the spatial and temporal dependencies in an image after it passes through a number of filters.

The first layer the input image passes through is a convolution layer. In mathematics, a convolution is an ordered method where two sources of information are connected to each other. A smaller matrix known as the kernel are convolved with the input value to obtain activation maps. The function of these activated maps is to indicate specific features of the image that are relevant to the kernel. A dot product is computed between the input matrix and the kernel to obtain a convolved value which structures a single entry in the activation matrix. The square selected is then slid over to the next patch and the process is repeated until the activation matrix is full (Raval,2019).


Blue Channel Matrix
Previous Patch (3x3)
Current Patch (3x3)
(Obtained after sliding from previous position by 1 unit, i.e. Stride = 1 unit)
Activation Map
Green Channel Matrix
Red Channel Matrix
Kernel (or Filter) Matrix (Sum of values = 2)

1	3	9	2
6	18	32	0
4	39	17	16
177	63	25	15

-9	-1	0
3	8	-6
5	2	0

56/2 + rcc1	503/2 + rcc2	

Figure 5 (Raval, 2019)

The next layer in the pipeline is the Pooling layer. The function of this layer is to reduce the spatial dimensions of the input to be processed by the next layer in the network. While this reduces the spatial dimension, it does not affect the depth dimension of the volume. It is achieved by taking the maximum value from the values observed. This technique is known as max pooling (Raval, 2019).

Negative numbers aren't suitable for this operation and can cause the math to break. To keep this from happening, the next layer known as the normalization layer turns all the negative numbers in the matrix into zeros. After this the convolutional layer and pooling layer are repeated several times until a smaller image with a new activation map is created at each iteration (Raval, 2019).

Once the features are learned by the neural network after repeated feature learning iterations, there are instances where the model might fit the training data too well and hence does not have a good prediction performance. To avoid this, dropout is used. Dropout creates scenarios where the artificial neural network is forced to learn multiple independent representations of the same data by randomly switching off neurons during the learning phase. To achieve this, some of the layer's values are randomly assigned to zero during forward propagation (Raval, 2019).

Finally, once the classification phase comes to an end, a SoftMax function is used to convert the outputs into binary values or probability values. A SoftMax function is a mathematical function that takes an input of real numbers and normalizes them into a probability distribution.

3.2.4 Dataset

Research on the use of thermal imaging as a screening method for breast cancer detection is not new. Several researchers have previously worked to prove the effectiveness of thermal imaging in addition to its main stream alternative, mammogram. The differences between a healthy breast as compared to a sick breast is very subtle but can be detected with the help of a trained model. A dataset of thermal images of patients was available online. A computing institute that specializes in the areas of computer vision and data visualization have collected such an image for a computer aided diagnosis system using a Support Vector Machine (SVM). As compared to a support vector machine, a convolutional neural network is far more advanced. SVM relies on linear classification whereas a CNN relies on non linear classification. The complexity of classification is increased greatly due to the number of layers added by the convolutional neural network.

In this data collection process, the thermal images were captured at intervals of 15 seconds and a total of 20 images were captured. The patients ranged from age 29 to 85 with a total of 43 Healthy patients and 24 patients diagnosed with breast cancer (Elias, 2019).

3.3 Design Diagram

3.3.1 Dataset Structure

The data file itself came organized as healthy and sick patients with folders for each patient. inside the patient folders there were 20 thermal images stored. Majority of these images resembled each other which might cause an issue for the neural network later.

Before preparing the classification model, the data needed to be organized. Besides this, the number of thermal images had to be reduced due to the similarity of the images. Finally, the images were classified into three folders namely, Training, Testing and Validation.

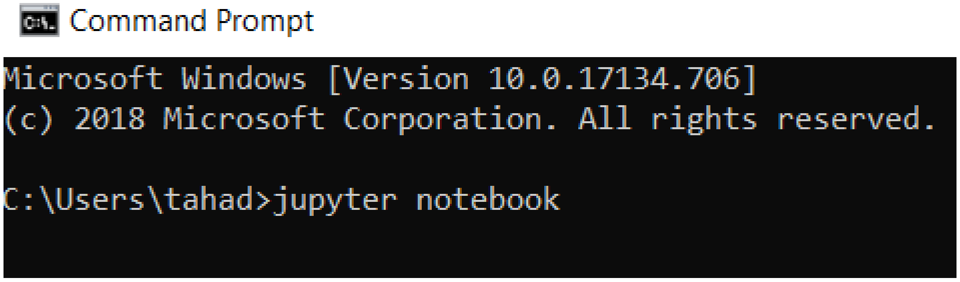
	Healthy	Sick
Training	250	250
Testing	100	100
Validation	10	10

Table 5

Chapter 4

Implementation

This work done on this project is conducted in Jupyter notebook. Jupyter notebook is a scientific notebook that allows users to work with python and code block by block which works very well for machine learning projects.

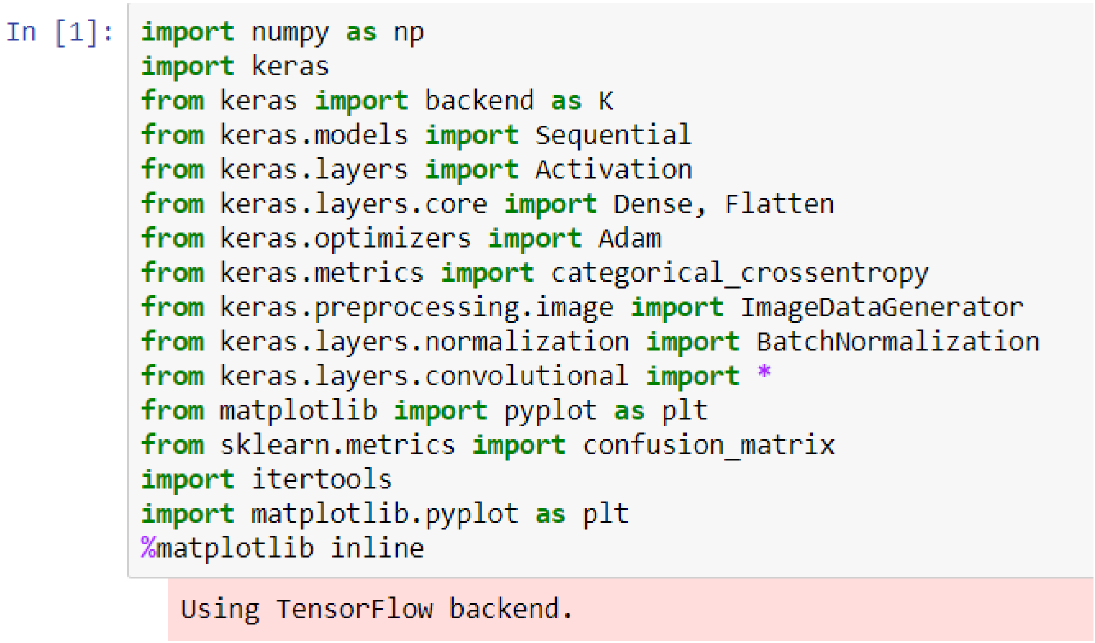


```
Microsoft Windows [Version 10.0.17134.706]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\tahad>jupyter notebook
```

Figure 6

To open Jupyter notebook, the command line is opened and “Jupyter notebook” is typed in to open up a browser where a new notebook can be opened.

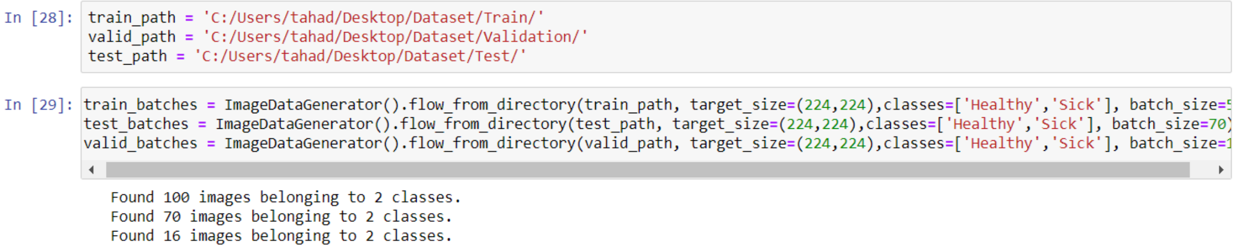
Once a new notebook is opened up, the imports necessary for this project are imported in the first cell so that these functions can be used in the cells below.



```
In [1]: import numpy as np
        import keras
        from keras import backend as K
        from keras.models import Sequential
        from keras.layers import Activation
        from keras.layers.core import Dense, Flatten
        from keras.optimizers import Adam
        from keras.metrics import categorical_crossentropy
        from keras.preprocessing.image import ImageDataGenerator
        from keras.layers.normalization import BatchNormalization
        from keras.layers.convolutional import *
        from matplotlib import pyplot as plt
        from sklearn.metrics import confusion_matrix
        import itertools
        import matplotlib.pyplot as plt
        %matplotlib inline

           Using TensorFlow backend.
```

Figure 7

Numpy, keras, sklearn, itertools and matplotlib are imported along with specific modules from these packages. Sklearn stands for sci-kit learn, it's a library that allows the user to import effective tools for data mining and analysis. The foundation for this library is from NumPy, SciPy and matplotlib. The libraries imported from keras allow us to create the various layers for the convolutional neural network and fine tune them to our requirement.

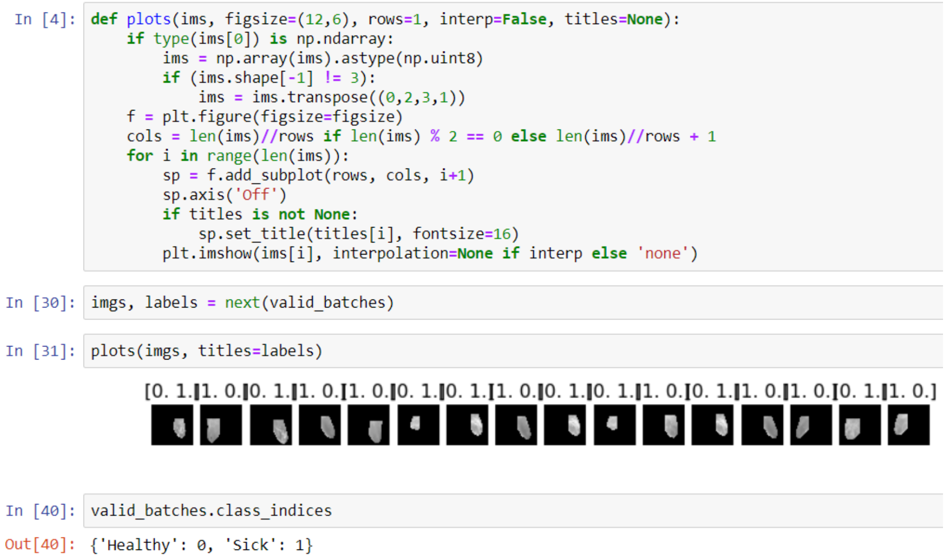


```
In [28]: train_path = 'C:/Users/tahad/Desktop/Dataset/Train/'
         valid_path = 'C:/Users/tahad/Desktop/Dataset/Validation/'
         test_path = 'C:/Users/tahad/Desktop/Dataset/Test/'

In [29]: train_batches = ImageDataGenerator().flow_from_directory(train_path, target_size=(224,224),classes=['Healthy','Sick'], batch_size=5
         test_batches = ImageDataGenerator().flow_from_directory(test_path, target_size=(224,224),classes=['Healthy','Sick'], batch_size=70)
         valid_batches = ImageDataGenerator().flow_from_directory(valid_path, target_size=(224,224),classes=['Healthy','Sick'], batch_size=1

           Found 100 images belonging to 2 classes.
           Found 70 images belonging to 2 classes.
           Found 16 images belonging to 2 classes.
```

Figure 8

The file path for the data needs to be specified for the image to be analyzed by the keras functions. In the cell below, the file path is specified to ImageDataGenerator() to generate batches of tensor image data which is the format the images need to be in to be read by the keras model. This is a class provided by keras that defines the configuration for preparation of image data and augmentation. The API is designed to reduce the memory overhead by being iterated by the deep learning model fitting process rather than performing operations on the entire image dataset in memory. Instead, it adds additional time during model training (Brownlee, 2019). The path for the training, testing and validation folder is specified along with the target size of the image and size of each batch.

To verify if the images are being loaded to Jupyter notebooks, they need to be printed with their respective labels. A python function Plots is written to print out the loaded images with labels on top of them.

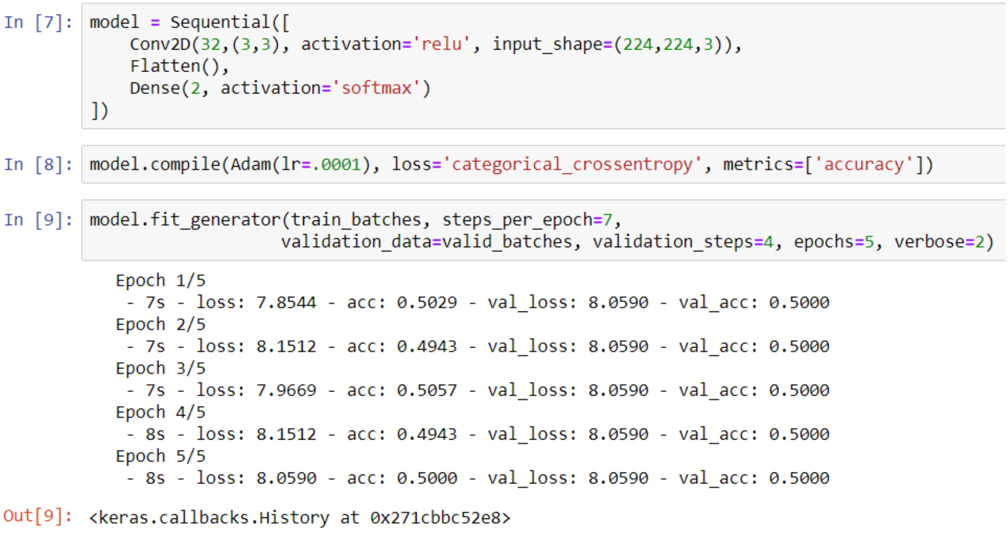


```
In [4]: def plots(ims, figsize=(12,6), rows=1, interp=False, titles=None):
            if type(ims[0]) is np.ndarray:
                ims = np.array(ims).astype(np.uint8)
                if (ims.shape[-1] != 3):
                    ims = ims.transpose((0,2,3,1))
            f = plt.figure(figsize=figsize)
            cols = len(ims)//rows if len(ims) % 2 == 0 else len(ims)//rows + 1
            for i in range(len(ims)):
                sp = f.add_subplot(rows, cols, i+1)
                sp.axis('Off')
                if titles is not None:
                    sp.set_title(titles[i], fontsize=16)
                plt.imshow(ims[i], interpolation=None if interp else 'none')

In [30]: imgs, labels = next(valid_batches)

In [31]: plots(imgs, titles=labels)
```

```
In [40]: valid_batches.class_indices

Out[40]: {'Healthy': 0, 'Sick': 1}
```

Figure 9

The images are plotted along with their labels confirming that the images are being loaded to Jupyter notebooks. The healthy images are classified as [1,0] and the sick images are classified as [0,1]. This is deducted as the class indices indicate that the index position of the binary marker.



```
In [7]: model = Sequential([
            Conv2D(32,(3,3), activation='relu', input_shape=(224,224,3)),
            Flatten(),
            Dense(2, activation='softmax')
        ])

In [8]: model.compile(Adam(lr=.0001), loss='categorical_crossentropy', metrics=['accuracy'])

In [9]: model.fit_generator(train_batches, steps_per_epoch=7,
                            validation_data=valid_batches, validation_steps=4, epochs=5, verbose=2)

           Epoch 1/5
            - 7s - loss: 7.8544 - acc: 0.5029 - val_loss: 8.0590 - val_acc: 0.5000
           Epoch 2/5
            - 7s - loss: 8.1512 - acc: 0.4943 - val_loss: 8.0590 - val_acc: 0.5000
           Epoch 3/5
            - 7s - loss: 7.9669 - acc: 0.5057 - val_loss: 8.0590 - val_acc: 0.5000
           Epoch 4/5
            - 8s - loss: 8.1512 - acc: 0.4943 - val_loss: 8.0590 - val_acc: 0.5000
           Epoch 5/5
            - 8s - loss: 8.0590 - acc: 0.5000 - val_loss: 8.0590 - val_acc: 0.5000

Out[9]: <keras.callbacks.History at 0x271cbbc52e8>
```

Figure 10

A sequential model is made with a single convolutional layer, a flatten layer which takes the output from the previous layer and flattens it into a 1 dimension tensor and a Dense layer. A dense layer is a regular layer of neurons in the neural network. We use the SoftMax activation as the endpoint of the Dense layer. Next the model is compiled with an Adam optimization function. It is an optimization algorithm that can be used to update the network weights. The loss is specified as categorical cross entropy and accuracy is used a metric. Finally, model.fit_generator is called to train the model. As expected, the accuracy produced is very poor as this isn't an ideal convolutional neural network. A good neural network consists of many layers and is previously trained extensively.

To solve this problem, Transfer learning is applied. Transfer learning is a widely used approach in machine learning where a model trained previously is applied as the starting point for another task. There are plenty of pre-trained models which have been trained by experts in the field. These pre trained models can be imported and fine tuned to produce the desirable classification required. For this project, two pre trained models will be experimented with to examine which one yields better results.

The first pre trained model to be used is known as VGG16. It is a convolutional neural network for large scale image recognition. When tested, the model achieved 92.7% accuracy on ImageNet, a dataset of over 14 million images. It was suggested by A. Zisserman and K. Simonyan from University of Oxford (Brownlee, 2019).

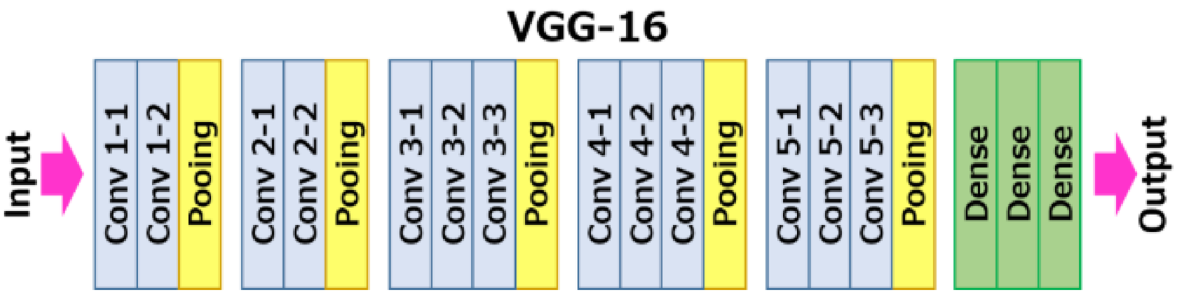
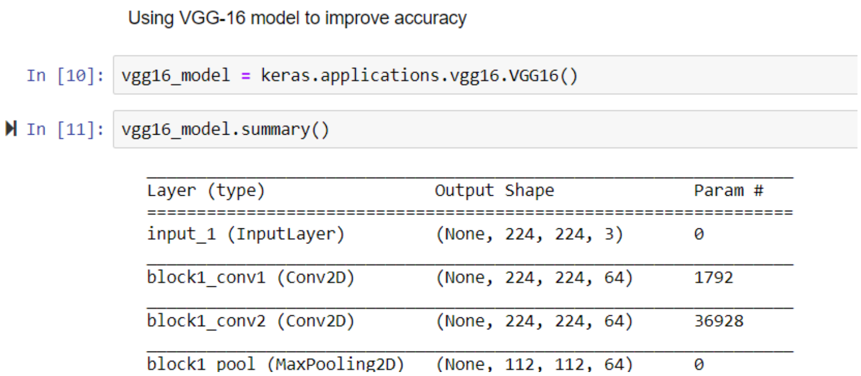


Figure 11 (Neurohive.io, 2019)

It improves upon its predecessor AlexNet by replacing the kernel-sized filters, 11 and 5 in the first and second layer with multiple 3 by 3 kernel sized filters one after the other. The model was trained for weeks and used NVIDIAs titan black GPU (Brownlee, 2019).



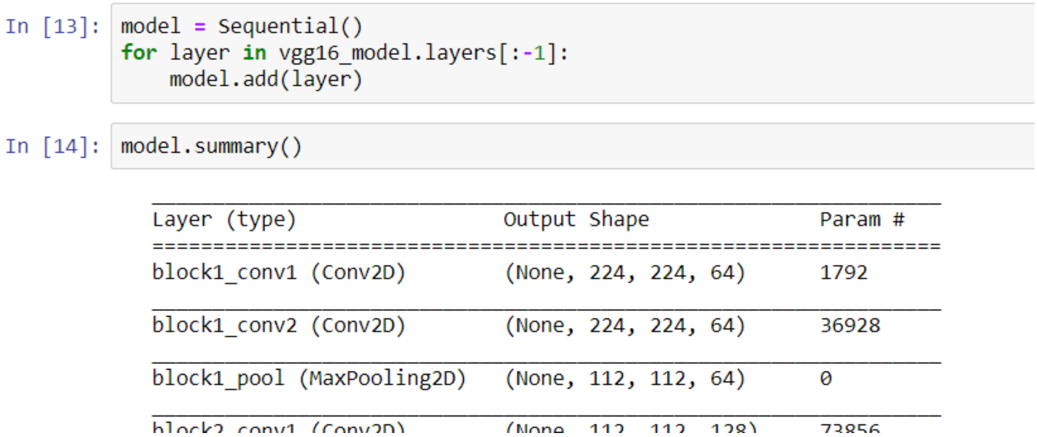
Using VGG-16 model to improve accuracy

```
In [10]: vgg16_model = keras.applications.vgg16.VGG16()

In [11]: vgg16_model.summary()
```

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         (None, 224, 224, 3)       0
_________________________________________________________________
block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
_________________________________________________________________
block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
_________________________________________________________________
block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
```

Figure 12

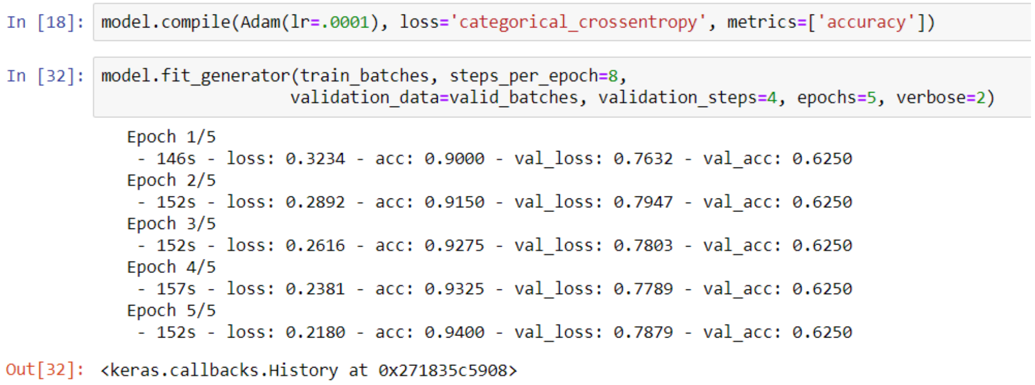
The VGG16 model is loaded to Jupyter notebook using keras and the summary of the model is viewed.



```
In [13]: model = Sequential()
         for layer in vgg16_model.layers[:-1]:
             model.add(layer)

In [14]: model.summary()

         _________________________________________________________________
         Layer (type)                 Output Shape              Param #
         =================================================================
         block1_conv1 (Conv2D)        (None, 224, 224, 64)      1792
         _________________________________________________________________
         block1_conv2 (Conv2D)        (None, 224, 224, 64)      36928
         _________________________________________________________________
         block1_pool (MaxPooling2D)   (None, 112, 112, 64)      0
         _________________________________________________________________
         block2_conv1 (Conv2D)        (None, 112, 112, 128)     73856
```

Figure 13

The VGG16 model is of type model rather than of type sequential. So, the model is transformed into a sequential model and all the layers in the VGG16 model is iterated and the layers are added to the new model created.



```
In [18]: model.compile(Adam(lr=.0001), loss='categorical_crossentropy', metrics=['accuracy'])

In [32]: model.fit_generator(train_batches, steps_per_epoch=8,
                             validation_data=valid_batches, validation_steps=4, epochs=5, verbose=2)

         Epoch 1/5
          - 146s - loss: 0.3234 - acc: 0.9000 - val_loss: 0.7632 - val_acc: 0.6250
         Epoch 2/5
          - 152s - loss: 0.2892 - acc: 0.9150 - val_loss: 0.7947 - val_acc: 0.6250
         Epoch 3/5
          - 152s - loss: 0.2616 - acc: 0.9275 - val_loss: 0.7803 - val_acc: 0.6250
         Epoch 4/5
          - 157s - loss: 0.2381 - acc: 0.9325 - val_loss: 0.7789 - val_acc: 0.6250
         Epoch 5/5
          - 152s - loss: 0.2180 - acc: 0.9400 - val_loss: 0.7879 - val_acc: 0.6250

Out[32]: <keras.callbacks.History at 0x271835c5908>
```

Figure 14

As done previously, the model is compiled, and the training begins. Unfortunately, when the original dataset size of 500 images was used, the kernel crashed, and the operation could not be completed. This is because VGG16 is a model that is very slow to train and requires high computational power. This is caused due to its depth and fully connected nodes. Due to this, the Dataset size was reduced, and the model was trained with insufficient data.

The next model to be tested for transfer learning is the Inception V3 Model. This model consists of 42 layers and resulted as a runner up in ImageNet. It resulted in accuracy greater than 78.1% and is improving constantly. It was developed by a team from google.

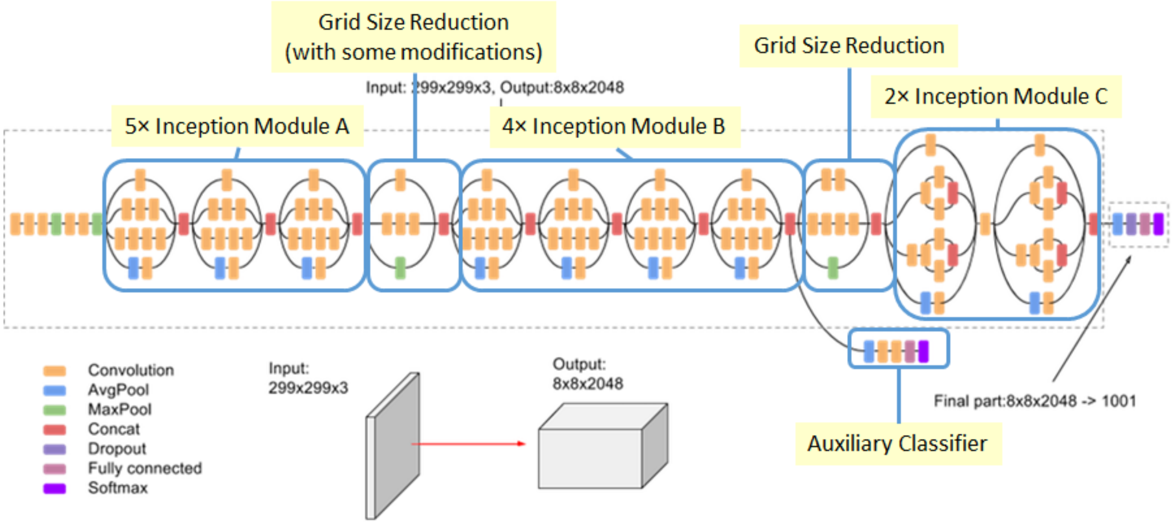
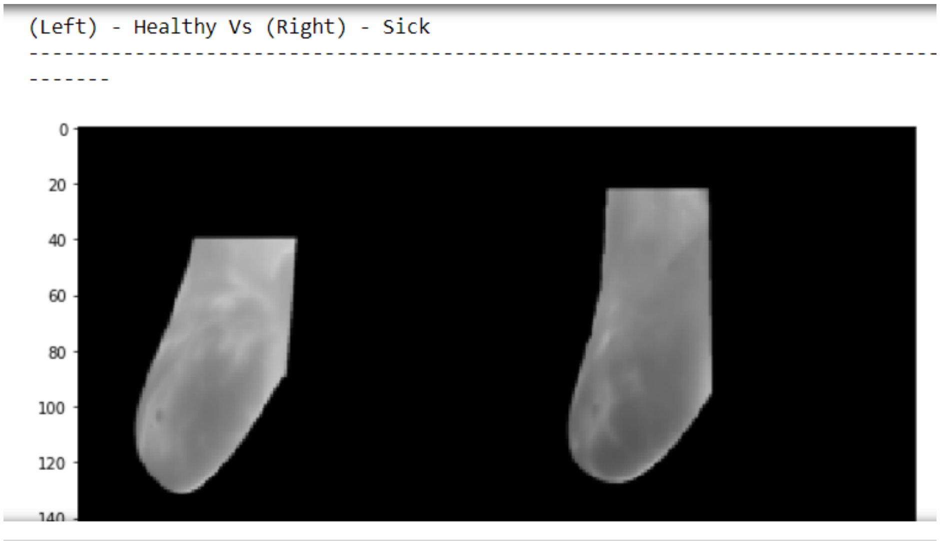


Figure 15 (Medium, 2019)

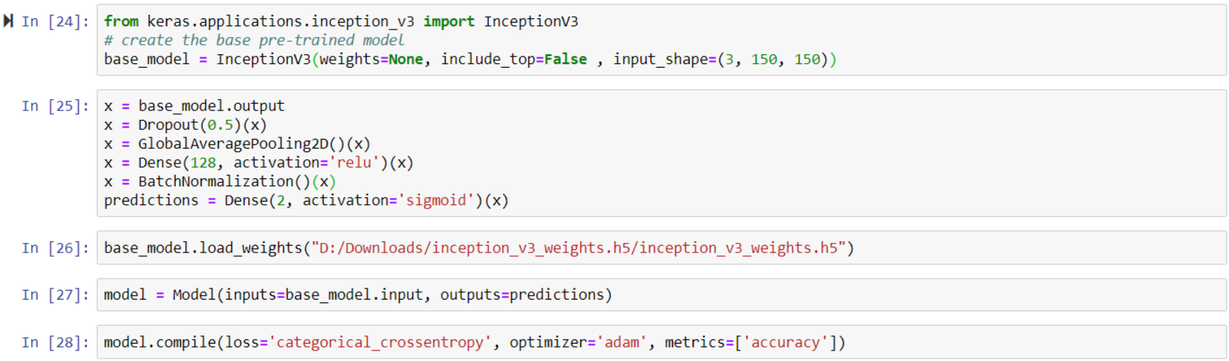
A new Jupyter notebook is opened and the original dataset with 500 images is loaded. A comparison is plotted to notice the differences between a healthy patient and a sick patient.



```
(Left) - Healthy Vs (Right) - Sick
---------------------------------------------------------------------------
-------
```

Figure 16

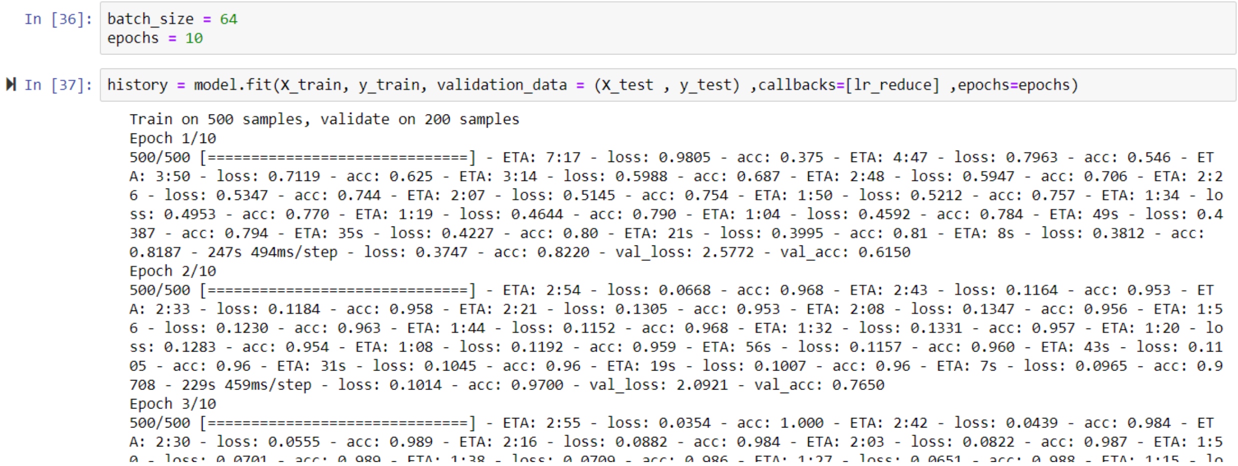
While the differences are very subtle, it can be noticed that the infrared image of the healthy breast displays more veins and in general is clearer than the sick patient. The infrared image of the breast with cancer displays a cloudy grey pattern than can be detected when iterated over many images.



```
In [24]: from keras.applications.inception_v3 import InceptionV3
         # create the base pre-trained model
         base_model = InceptionV3(weights=None, include_top=False , input_shape=(3, 150, 150))

In [25]: x = base_model.output
         x = Dropout(0.5)(x)
         x = GlobalAveragePooling2D()(x)
         x = Dense(128, activation='relu')(x)
         x = BatchNormalization()(x)
         predictions = Dense(2, activation='sigmoid')(x)

In [26]: base_model.load_weights("D:/Downloads/inception_v3_weights.h5/inception_v3_weights.h5")

In [27]: model = Model(inputs=base_model.input, outputs=predictions)

In [28]: model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
```

Figure 17

The inception V3 model is imported and a few new layers are added for better accuracy. The base weights from the inception V3 are loaded and the model is compiled similarly to the way it was compiled for the VGG 16 model.



```
In [36]: batch_size = 64
         epochs = 10

In [37]: history = model.fit(X_train, y_train, validation_data = (X_test , y_test) ,callbacks=[lr_reduce] ,epochs=epochs)

    Train on 500 samples, validate on 200 samples
    Epoch 1/10
    500/500 [==============================] - ETA: 7:17 - loss: 0.9805 - acc: 0.375 - ETA: 4:47 - loss: 0.7963 - acc: 0.546 - ET
    A: 3:50 - loss: 0.7119 - acc: 0.625 - ETA: 3:14 - loss: 0.5988 - acc: 0.687 - ETA: 2:48 - loss: 0.5947 - acc: 0.706 - ETA: 2:2
    6 - loss: 0.5347 - acc: 0.744 - ETA: 2:07 - loss: 0.5145 - acc: 0.754 - ETA: 1:50 - loss: 0.5212 - acc: 0.757 - ETA: 1:34 - lo
    ss: 0.4953 - acc: 0.770 - ETA: 1:19 - loss: 0.4644 - acc: 0.790 - ETA: 1:04 - loss: 0.4592 - acc: 0.784 - ETA: 49s - loss: 0.4
    387 - acc: 0.794 - ETA: 35s - loss: 0.4227 - acc: 0.80 - ETA: 21s - loss: 0.3995 - acc: 0.81 - ETA: 8s - loss: 0.3812 - acc:
    0.8187 - 247s 494ms/step - loss: 0.3747 - acc: 0.8220 - val_loss: 2.5772 - val_acc: 0.6150
    Epoch 2/10
    500/500 [==============================] - ETA: 2:54 - loss: 0.0668 - acc: 0.968 - ETA: 2:43 - loss: 0.1164 - acc: 0.953 - ET
    A: 2:33 - loss: 0.1184 - acc: 0.958 - ETA: 2:21 - loss: 0.1305 - acc: 0.953 - ETA: 2:08 - loss: 0.1347 - acc: 0.956 - ETA: 1:5
    6 - loss: 0.1230 - acc: 0.963 - ETA: 1:44 - loss: 0.1152 - acc: 0.968 - ETA: 1:32 - loss: 0.1331 - acc: 0.957 - ETA: 1:20 - lo
    ss: 0.1283 - acc: 0.954 - ETA: 1:08 - loss: 0.1192 - acc: 0.959 - ETA: 56s - loss: 0.1157 - acc: 0.960 - ETA: 43s - loss: 0.11
    05 - acc: 0.96 - ETA: 31s - loss: 0.1045 - acc: 0.96 - ETA: 19s - loss: 0.1007 - acc: 0.96 - ETA: 7s - loss: 0.0965 - acc: 0.9
    708 - 229s 459ms/step - loss: 0.1014 - acc: 0.9700 - val_loss: 2.0921 - val_acc: 0.7650
    Epoch 3/10
    500/500 [==============================] - ETA: 2:55 - loss: 0.0354 - acc: 1.000 - ETA: 2:42 - loss: 0.0439 - acc: 0.984 - ET
    A: 2:30 - loss: 0.0555 - acc: 0.989 - ETA: 2:16 - loss: 0.0882 - acc: 0.984 - ETA: 2:03 - loss: 0.0822 - acc: 0.987 - ETA: 1:5
    0 - loss: 0.0701 - acc: 0.989 - ETA: 1:38 - loss: 0.0709 - acc: 0.986 - ETA: 1:27 - loss: 0.0651 - acc: 0.988 - ETA: 1:15 - lo
```

Figure 18

Next the batch size and epochs are specified right before the model is set to train. Instantly, a very vivid difference is noted as compared to the VGG16 model. The inception V3 model is much smoother and trains without draining computational power from the device.

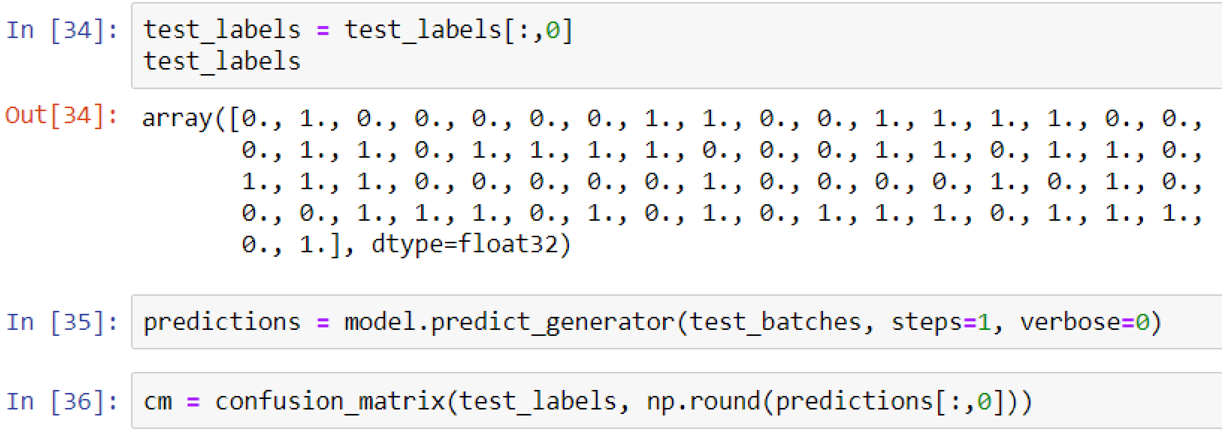
Chapter 5

Evaluation

To test how well the model performs with new data, a confusion matrix will be plotted. A confusion matrix is a table that is used to describe the performance and accuracy of a classification model. A table is plotted from which we can calculate the true positives, true negatives, false positives and false negatives. Besides the confusion matrix, we can calculate certain metrics that are used to indicate how well a model classifies the data provided to it. The metrics that are going to be used are Precision, recall and f1 score.

Precision and recall are very straight forward and yet powerful ways to measure the quality of predictions. Precision is the number of true positives divided by the number of true positives plus the number of false positives. On the other hand, recall is the number of true positives divided by the number of true positives plus the number of false negatives.

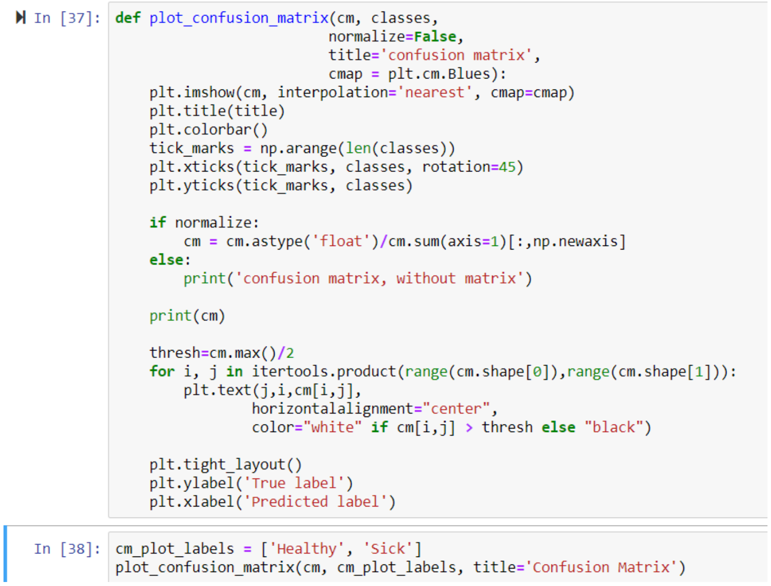
According to some researchers, the f1 score might be the better metric to measure the success of the model’s prediction. It is considered as the balance between precision and recall. It is calculated by dividing the precision multiplied by the recall with the precision added to the recall and multiplying the output by 2.



```
In [34]: test_labels = test_labels[:,0]
         test_labels

Out[34]: array([0., 1., 0., 0., 0., 0., 0., 1., 1., 0., 0., 1., 1., 1., 1., 0., 0.,
                0., 1., 1., 0., 1., 1., 1., 1., 0., 0., 0., 1., 1., 0., 1., 1., 0.,
                1., 1., 1., 0., 0., 0., 0., 0., 1., 0., 0., 0., 0., 1., 0., 1., 0.,
                0., 0., 1., 1., 1., 0., 1., 0., 1., 0., 1., 1., 1., 0., 1., 1., 1.,
                0., 1.], dtype=float32)

In [35]: predictions = model.predict_generator(test_batches, steps=1, verbose=0)

In [36]: cm = confusion_matrix(test_labels, np.round(predictions[:,0]))
```

Figure 19

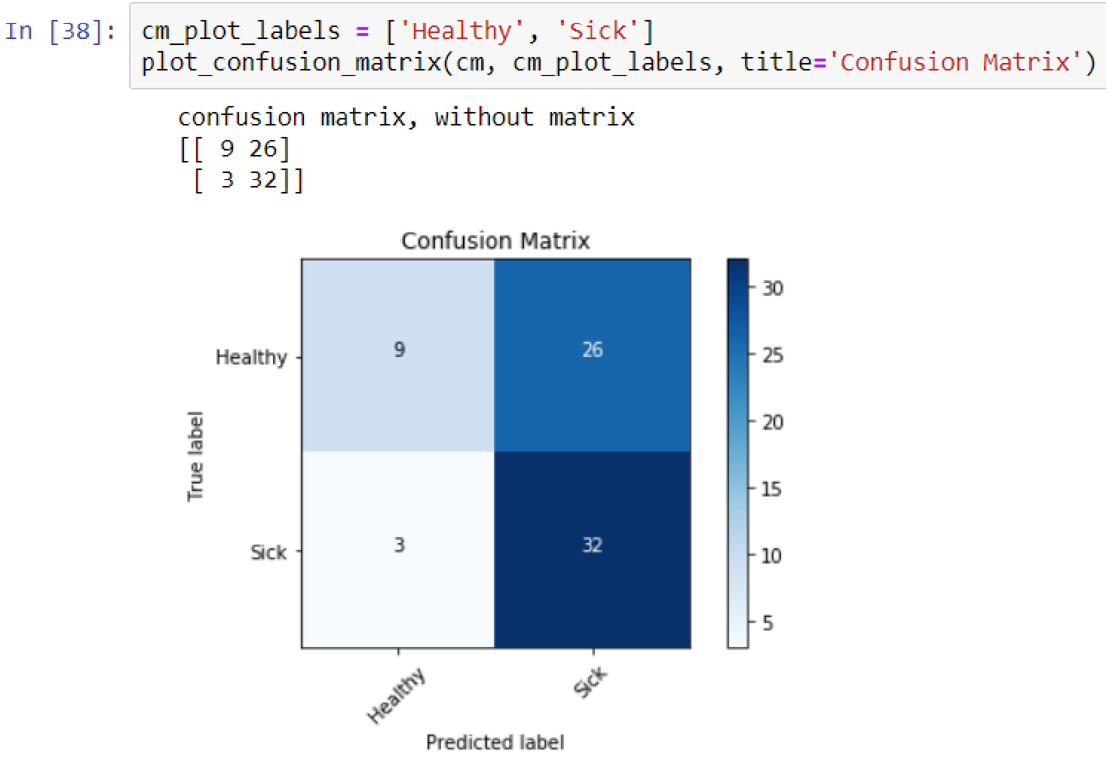
To plot the confusion matrix for the VGG16 model, the test data needs to be prepared. To do so, the labels are first split to print only the zero index value to get binary classification. Then the confusion matrix is assigned the real test labels and the predicted test labels.



```
In [37]: def plot_confusion_matrix(cm, classes,
                                   normalize=False,
                                   title='confusion matrix',
                                   cmap = plt.cm.Blues):
             plt.imshow(cm, interpolation='nearest', cmap=cmap)
             plt.title(title)
             plt.colorbar()
             tick_marks = np.arange(len(classes))
             plt.xticks(tick_marks, classes, rotation=45)
             plt.yticks(tick_marks, classes)

             if normalize:
                 cm = cm.astype('float')/cm.sum(axis=1)[:,np.newaxis]
             else:
                 print('confusion matrix, without matrix')

             print(cm)

             thresh=cm.max()/2
             for i, j in itertools.product(range(cm.shape[0]),range(cm.shape[1])):
                 plt.text(j,i,cm[i,j],
                         horizontalalignment="center",
                         color="white" if cm[i,j] > thresh else "black")

             plt.tight_layout()
             plt.ylabel('True label')
             plt.xlabel('Predicted label')

In [38]: cm_plot_labels = ['Healthy', 'Sick']
         plot_confusion_matrix(cm, cm_plot_labels, title='Confusion Matrix')
```

Figure 20

A python function is written to plot the confusion matrix, it assigns colors, design and the true and predicted labels to the confusion matrix.



```
In [38]: cm_plot_labels = ['Healthy', 'Sick']
         plot_confusion_matrix(cm, cm_plot_labels, title='Confusion Matrix')
```

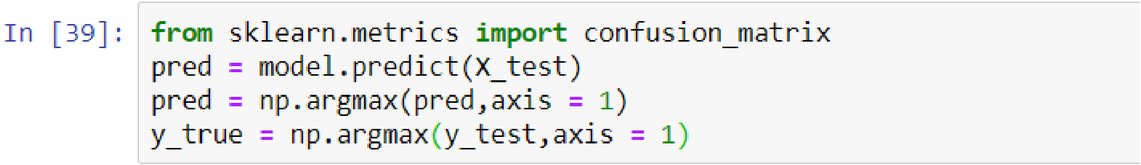
```
confusion matrix, without matrix
[[ 9 26]
 [ 3 32]]
```

Figure 21

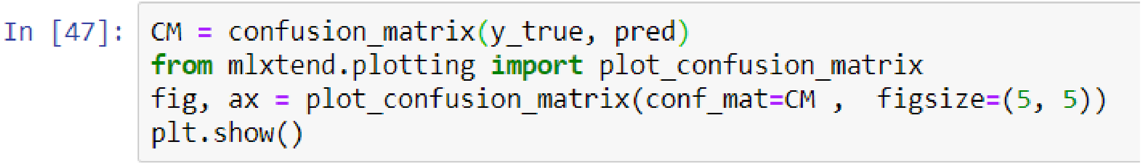
The resulting confusion matrix concludes that the number of healthy patients predicted accurately were 9 and the number healthy patients predicted as sick were 26. The number of sick patients predicted accurately were 32 and the number of sick patients predicted as healthy were 3. While on the overview it may seem like the model is accurately predicting the number of sick patients, on a closer inspection it is noted that the model predicts a larger number of patients as sick and only a few patients as healthy.

This confusion matrix produces a false positive of 26, a false negative of 3, a true positive of 9 and a true negative of 32. According to these metrics, a precision of 25% is calculated, a recall of 75% and an f1 score of 0.37. These results are considered as very poor and cannot be used to predict real life cases.

Next, we move on to the results of using the Inception V3 model.



```
In [39]: from sklearn.metrics import confusion_matrix
         pred = model.predict(X_test)
         pred = np.argmax(pred,axis = 1)
         y_true = np.argmax(y_test,axis = 1)
```



```
In [47]: CM = confusion_matrix(y_true, pred)
         from mlxtend.plotting import plot_confusion_matrix
         fig, ax = plot_confusion_matrix(conf_mat=CM ,  figsize=(5, 5))
         plt.show()
```

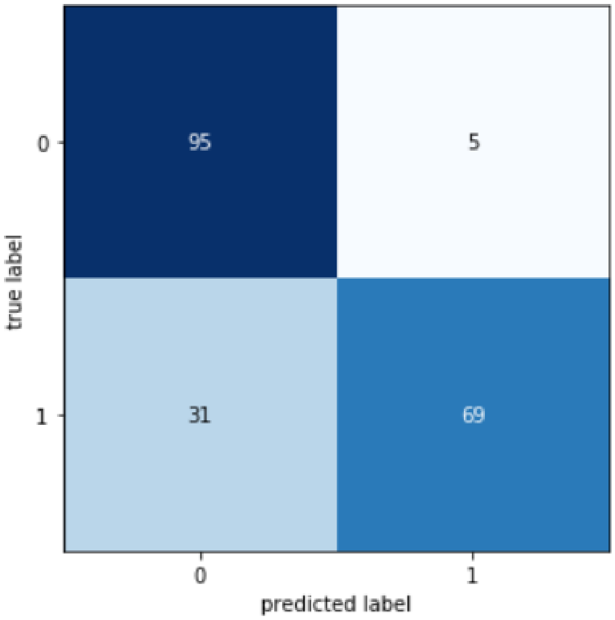


Figure 22

When the confusion matrix for the Inception v3 model is plotted, the results in a quick overview are considerably much better as compared to the VGG model. The healthy images are labelled as zero and the sick images are labelled as 1 in this confusion matrix. The confusion matrix displays a true positive of 95, a true negative of 69, a false positive of 5 and a false negative of 31. According to these numbers, the precision can be calculated as 95% which is a massive leap as compared to the previous precision score. The recall can be calculated as 75% and the f1 score can be calculated as 0.83.

Chapter 6

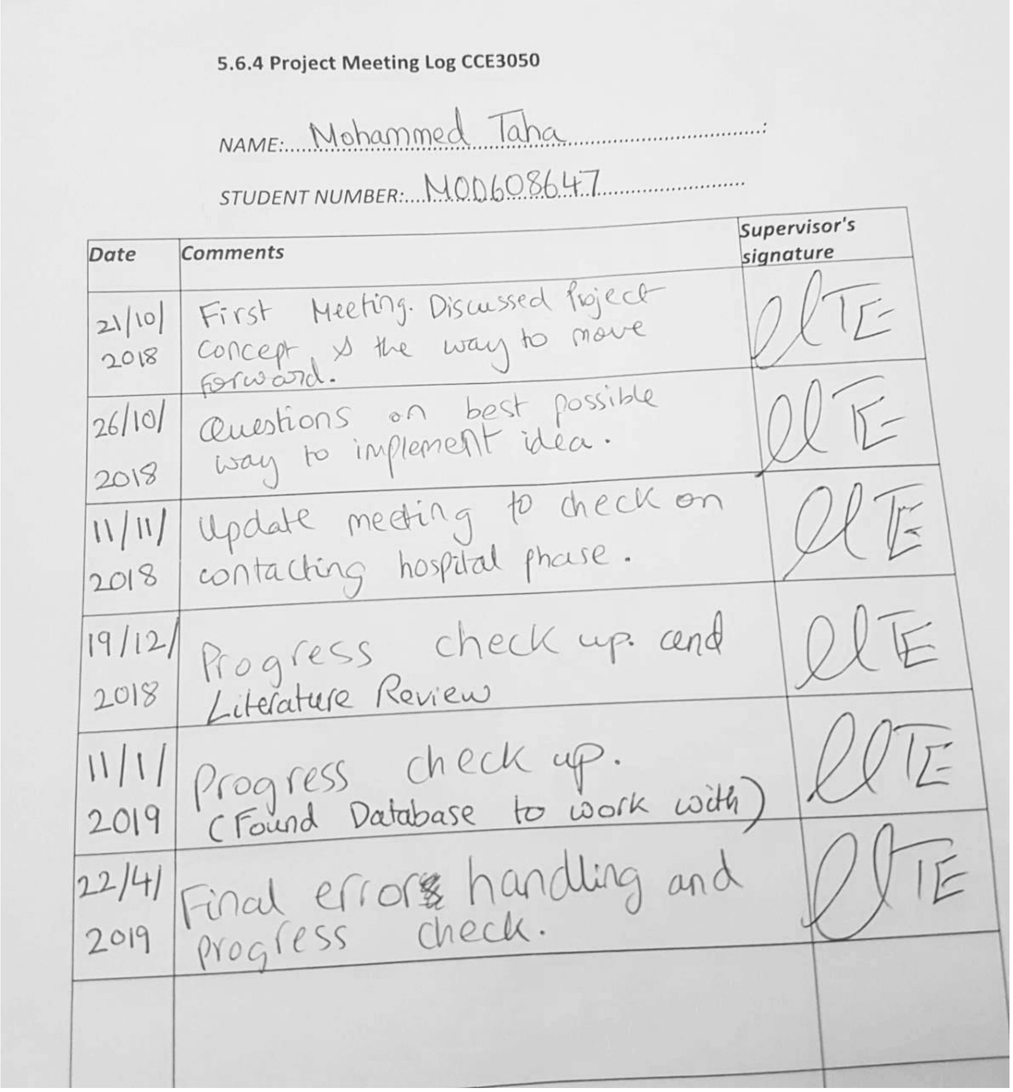
Conclusion

In this paper, Research is conducted on the effectiveness of Thermal infrared imaging for breast cancer detection over the main stream alternative of Mammograms. Mammograms have been reported as an invasive and expensive testing method. Women in low income countries tend to avoid getting themselves tested for breast cancer every 2 years and this decreases the survival rate due to lack of awareness and early checks. Mammograms have also been accurate all the time. According to the American cancer society, a large number of false negatives and false positives have been reported about mammograms. The alternative to mammograms is a non invasive method that can detect the growth of a tumor much earlier than mammograms. The topic of using thermography as a diagnosis tool is constantly being researched by countless researchers. Besides research, the method is even being used in the United States as an alternative to mammograms at a few health centres such as the O2 Wellness Centre. According to these papers and institutes, Thermography can be used for prediction of cardiovascular diseases, peripheral vascular disorders, measurement of human vital signs, early indication of breast cancer and much more. This is a considerable amount of proof that thermography is an effective yet underrated technique.

In conclusion, the Inception v3 model proves to be considerably very accurate. With a precision of 95%, such a model can be improved upon with access to more training data. A dataset of only 500 images is considered as a really small dataset for a convolutional neural network which has been created by training and testing on millions of images. With access to a larger dataset and medical assistance on this project, the number of false positives and negatives can be decreased even more.

Appendices

Appendix A: Meeting log sheet



5.6.4 Project Meeting Log CCE3050

NAME: Mohammed Taha

STUDENT NUMBER: M00608647

Date	Comments	Supervisor's signature
21/10/2018	First Meeting. Discussed Project concept & the way to move forward.	
26/10/2018	Questions on best possible way to implement idea.	
11/11/2018	Update meeting to check on contacting hospital phase.	
19/12/2018	Progress check up. and Literature Review	
11/1/2019	Progress check up. (Found Database to work with)	
22/4/2019	Final errors handling and progress check.	

Appendix B: Code

```
import pandas as pd

import cv2

import numpy as np

import os

from random import shuffle

from tqdm import tqdm

import scipy

import skimage

from skimage.transform import resize

print(os.listdir("D:/Breast_Cancer_Detection/Infrared"))


TRAIN_DIR = "D:/Breast_Cancer_Detection/Infrared/Train/"

TEST_DIR = "D:/Breast_Cancer_Detection/Infrared/Test/"


def get_label(Dir):

    for nextdir in os.listdir(Dir):

        if not nextdir.startswith('.'):

            if nextdir in ['Healthy']:

                label = 0

            elif nextdir in ['Sick']:

                label = 1

            else:

                label = 2

    return nextdir, label
```

```
def preprocessing_data(Dir):
    X = []
    y = []

    for nextdir in os.listdir(Dir):
        nextdir, label = get_label(Dir)
        temp = Dir + nextdir

        for image_filename in tqdm(os.listdir(temp)):
            path = os.path.join(temp + '/' , image_filename)
            img = cv2.imread(path,cv2.IMREAD_GRAYSCALE)
            if img is not None:
                img = skimage.transform.resize(img, (150, 150, 3))
                img = np.asarray(img)
                X.append(img)
                y.append(label)

    X = np.asarray(X)
    y = np.asarray(y)

    return X,y
```

```
def get_data(Dir):
    X = []
    y = []
    for nextDir in os.listdir(Dir):
        if not nextDir.startswith('.'):
            if nextDir in ['Healthy']:
                label = 0
            elif nextDir in ['Sick']:
                label = 1
            else:
                label = 2

            temp = Dir + nextDir

            for file in tqdm(os.listdir(temp)):
                img = cv2.imread(temp + '/' + file)
                if img is not None:
                    img = skimage.transform.resize(img, (150, 150, 3))
                    #img_file = scipy.misc.imresize(arr=img_file, size=(150, 150, 3))
                    img = np.asarray(img)
                    X.append(img)
                    y.append(label)

    X = np.asarray(X)
    y = np.asarray(y)
    return X,y
```

```
X_train, y_train = get_data(TRAIN_DIR)


X_test, y_test = get_data(TEST_DIR)


from keras.utils.np_utils import to_categorical


y_train = to_categorical(y_train, 2)
y_test = to_categorical(y_test, 2)


Himages = os.listdir(TRAIN_DIR + "Healthy")
Simages = os.listdir(TRAIN_DIR + "Sick")


from keras.models import Sequential , Model
from keras.layers import Dense , Activation
from keras.layers import Dropout , GlobalAveragePooling2D
from keras.layers import Flatten
from keras.constraints import maxnorm
from keras.optimizers import SGD , RMSprop , Adadelta , Adam
from keras.layers import Conv2D , BatchNormalization
from keras.layers import MaxPooling2D
from keras.utils import np_utils
from keras import backend as K
K.set_image_dim_ordering('th')
from sklearn.model_selection import GridSearchCV
from keras.wrappers.scikit_learn import KerasClassifier
```

```
X_train=X_train.reshape(500,3,150,150)
X_test=X_test.reshape(200,3,150,150)

from keras.applications.inception_v3 import InceptionV3
# create the base pre-trained model
base_model = InceptionV3(weights=None, include_top=False , input_shape=(3, 150, 150))

x = base_model.output
x = Dropout(0.5)(x)
x = GlobalAveragePooling2D()(x)
x = Dense(128, activation='relu')(x)
x = BatchNormalization()(x)
predictions = Dense(2, activation='sigmoid')(x)

base_model.load_weights("D:/Downloads/inception_v3_weights.h5/inception_v3_weights.h5")

model = Model(inputs=base_model.input, outputs=predictions)
model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
batch_size = 64
epochs = 10
history = model.fit(X_train, y_train, validation_data = (X_test , y_test) ,callbacks=[lr_reduce]
,epochs=epochs)
```

```
import matplotlib.pyplot as plt


plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()
# summarize history for loss
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title('model loss')
plt.ylabel('loss')
plt.xlabel('epoch')
plt.legend(['train', 'test'], loc='upper left')
plt.show()


from sklearn.metrics import confusion_matrix
pred = model.predict(X_test)
pred = np.argmax(pred,axis = 1)
y_true = np.argmax(y_test,axis = 1)
CM = confusion_matrix(y_true, pred)
from mlxtend.plotting import plot_confusion_matrix
fig, ax = plot_confusion_matrix(conf_mat=CM ,  figsize=(5, 5))
plt.show()
```

References

1. Mayo Clinic. (2019). Breast cancer - Symptoms and causes. [online] Available at: https://www.mayoclinic.org/diseases-conditions/breast-cancer/symptoms-causes/syc-20352470 [Accessed 24 Apr. 2019].
2. Mukerji, C. (2019). Can you bear the cost of cancer treatment? Find out how to buy the best cover. [online] The Economic Times. Available at: https://economictimes.indiatimes.com/wealth/insure/can-you-bear-the-cost-of-cancer-treatment-find-out-how-to-buy-the-best-cover/articleshow/47744432.cms [Accessed 24 Apr. 2019].
3. Kandlikar, Satish G. et al. (2017) 'Infrared imaging technology for breast cancer detection – Current status, protocols and new directions', International Journal of Heat and Mass Transfer. Elsevier Ltd, 108(PB), pp. 2303–2320. doi: 10.1016/j.ijheatmasstransfer.2017.01.086.
4. Towards Data Science. (2019). Deep Learning & Medical Diagnosis. [online] Available at: https://towardsdatascience.com/deep-learning-medical-diagnosis-c04d35fc2830 [Accessed 24 Apr. 2019].
5. Health news, Medibulletin. (2019). Why Modicare isn't making waves in Haryana's "cancer village" - Health news, Medibulletin. [online] Available at: https://medibulletin.com/why-modicare-isnt-making-waves-in-haryanas-cancer-village/ [Accessed 24 Apr. 2019].
6. World Health Organization. (2019). Breast cancer: prevention and control. [online] Available at: https://www.who.int/cancer/detection/breastcancer/en/index1.html [Accessed 25 Apr. 2019].
7. Gupta, A., Shridhar, K. and Dhillon, P. (2015). A review of breast cancer awareness among women in India: Cancer literate or awareness deficit? European Journal of Cancer, 51(14), pp.2058-2066.
8. Gupta, S. and Tripathi, P. (2016). "Big data lakes can support better population health for rural India — Swastha Bharat," 2016 International Conference on Innovation and Challenges in Cyber Security (ICICCS-INBUSH), Noida, 2016, pp. 145-150. doi: 10.1109/ICICCS.2016.7542361
9. Frize, M., Herry, C. and Scales, N. (n.d.). Processing thermal images to detect breast cancer and assess pain. 4th International IEEE EMBS Special Topic Conference on Information Technology Applications in Biomedicine, 2003..
10. Chekmenev, S.Y. & Rara, Helen & Farag, A.A.. (2005). Non-contact, wavelet-based measurement of vital signs using thermal imaging. The First International Conference on Graphics, Vision, and Image Processing (GVIP). 107-112.
11. Sruthi, S. and Sasikala, M. (2015). A low cost thermal imaging system for medical diagnostic applications. 2015 International Conference on Smart Technologies and Management for Computing, Communication, Controls, Energy and Materials (ICSTM).
12. Roychowdhury, S., Koozekanani, D. and Parhi, K. (2014). DREAM: Diabetic Retinopathy Analysis Using Machine Learning. IEEE Journal of Biomedical and Health Informatics, 18(5), pp.1717-1728.
13. O2 Wellness Center. (2018). Medical Thermal Imaging - O2 Wellness Center. [online] Available at: http://o2wellness.com/medical-thermal-imaging/ [Accessed 25 Oct. 2018].
14. Perez, F. (2019). Project Jupyter. [online] Speaker Deck. Available at: https://speakerdeck.com/fperez/project-jupyter?slide=9 [Accessed 26 Apr. 2019].
15. Numpy.org. (2019). NumPy — NumPy. [online] Available at: http://www.numpy.org/ [Accessed 26 Apr. 2019].
16. Pandas.pydata.org. (2019). Python Data Analysis Library — pandas: Python Data Analysis Library. [online] Available at: https://pandas.pydata.org/ [Accessed 26 Apr. 2019].
17. Matplotlib.org. (2019). Matplotlib: Python plotting — Matplotlib 3.0.3 documentation. [online] Available at: https://matplotlib.org/ [Accessed 26 Apr. 2019].
18. Opencv.org. (2019). OpenCV Documentation. [online] Available at: https://opencv.org/about/ [Accessed 26 Apr. 2019].

19. Keras.io. (2019). Home - Keras Documentation. [online] Available at: https://keras.io/ [Accessed 26 Apr. 2019].
20. Ahire, J. (2019). The Artificial Neural Networks handbook: Part 1. [online] Medium. Available at: https://medium.com/coinmonks/the-artificial-neural-networks-handbook-part-1-f9ceb0e376b4 [Accessed 26 Apr. 2019].
21. Raval, S. (2019). llSourcell/Convolutional_neural_network. [online] GitHub. Available at: https://github.com/llSourcell/Convolutional_neural_network/blob/master/convolutional_network_tutorial.ipynb [Accessed 26 Apr. 2019].
22. Saha, S. (2019). *A Comprehensive Guide to Convolutional Neural Networks—the ELI5 way*. [online] Towards Data Science. Available at: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53 [Accessed 26 Apr. 2019].
23. Elias, T. (2019). *Visual Lab - A Methodology for Breast Disease Computer-Aided Diagnosis*. [online] Visual.ic.uff.br. Available at: http://visual.ic.uff.br/en/proeng/thiagoelias/ [Accessed 26 Apr. 2019].
24. Brownlee, J. (2019). *Image Augmentation for Deep Learning With Keras*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/image-augmentation-deep-learning-keras/ [Accessed 26 Apr. 2019].
25. Brownlee, J. (2019). *A Gentle Introduction to Transfer Learning for Deep Learning*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/transfer-learning-for-deep-learning/ [Accessed 26 Apr. 2019].
26. Neurohive.io. (2019). *VGG16 - Convolutional Network for Classification and Detection*. [online] Available at: https://neurohive.io/en/popular-networks/vgg16/ [Accessed 26 Apr. 2019].
27. Medium. (2019). *Review: Inception-v3 — 1st Runner Up (Image Classification) in ILSVRC 2015*. [online] Available at: https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c [Accessed 26 Apr. 2019].