

## ABSTRACT

Breast cancer has an important incidence in women worldwide. Early diagnosis of this disease plays a key role in decreasing its mortality and improves its prognosis. Currently, mammography is considered as the standard examination for detection of breast cancer. However, the identification of breast abnormalities and the classification of masses on mammographic images are not trivial tasks for dense breasts, and is a challenge for artificial intelligence and pattern recognition. This work presents preliminary results of automatic classification of mammography images using image processing techniques. We implement a deep learning architecture based on the inception v3 model for classification in two classes: benign and malignant. The set of images for training and testing the model, are taken from the Digital Database for Screening Mammography (DDSM). Results show that the percentage of correct classification occurs in average for 56% of the data set with a recall rate of 76%.

# Table of Contents

INTRODUCTION.....	6
1.1 What Is Breast Cancer?.....	6
1.2 Where breast cancer starts?.....	6
1.3 How Common Is Breast Cancer?.....	7
1.4 Current year estimates for breast cancer.....	7
1.5 Trends in breast cancer incidence.....	7
1.6 Trends in breast cancer deaths.....	7
1.7 Early Detection of Breast Cancer.....	8
1.8 Mammograms.....	8
1.9 Why breast screening is important?.....	9
1.10 Breast cancer detection technologies.....	9
1.10.1 Digital Mammography.....	9
1.10.2 Computer-Aided Detection (CAD).....	10
1.10.3 Ultrasound Imaging.....	11
1.10.4 Magnetic Resonance Imaging (MRI).....	12
TERMINOLOGIES.....	14
2.1 Convolutional neural network (ConvNet).....	14
2.2 Dilation.....	15
2.3 Image Scaling.....	15
2.4 Normalization.....	16
2.5 Image Resizing.....	16
METHODOLOGIES.....	17
3.1 Dataset Used.....	17
3.2 Data Preprocessing.....	18
3.3 Data Augmentation.....	19
3.4 Network Architecture.....	19
SOFTWARE TOOLS.....	21
4.1 Keras.....	21
4.2 Pandas.....	21
4.3 OpenCV.....	22
4.4 Numpy.....	22
4.5 Jupyter Notebook.....	23
4.6 Matplotlib.....	24
4.7 PyDICOM.....	25

4.8 Google’s Colaboratory.....	27
SOFTWARE IMPLEMENTATION.....	28
EXPERIMENTAL RESULTS.....	34
CONCLUSION.....	35
FUTURE SCOPE.....	36
REFERENCES.....	37

**List of figure:****Page No.**

Figure 1.1: Mammogram.....	8
Figure 1.2: Breast Cancer in Woman.....	9
Figure 1.3: Examples of Full-Field Digital Mammography of the breast.....	10
Figure 1.4: An example of an ultrasound image of the breast .....	11
Figure 1.5: Example of a magnetic resonance image of the breast.....	12
Figure 2.1: Convolutional neural network.....	14
Figure 2.2: Original Image (Dilation).....	15
Figure 2.3: After Dilation.....	15
Figure 3.1: Mammogram image.....	17
Figure 3.2: Microcalcifications zoomed.....	17
Figure 3.3: Image Processing Implementation.....	18
Figure 3.4: Original image (Augmentation).....	19
Figure 3.5: Augmented image (180° rotate over the original).....	19
Figure 3.6: Inception model (naïve version).....	20
Figure 4.1: Loading the CSV file as a Data frame object and printing the first 5 rows.....	22
Figure 4.2: Utilizing the tools of both OpenCV and Numpy.....	23
Figure 4.3: Notebooks which can be run to perform data analysis.....	24
Figure 4.4: Using Matplotlib to plot the region of interest image.....	25
Figure 4.5: First set of Data Attributes related to the mammogram can be seen above.....	26
Figure 4.6: Mammogram Image (ROI).....	26
Figure 4.7: Google Colaboratory notebook.....	27
Figure 5.1: Importing the necessary packages. Packages specified in the <b>Software Tools</b> section, detail what each module is used for.....	28
Figure 5.2: Loading the CSV file of the image database as a Pandas dataframe object and performing a check to see if all the images recorded in the file are present in the image folder. If not, they are ignored and a new CSV file is created, keeping track of only those images that are present.....	28
Figure 5.3: Defining the image preprocessing function.....	29
Figure 5.4: Data augmentation and the calling of the <i>preprocess</i> function.....	29
Figure 5.5: This line of code (that can be added into any notebook), loads the pickle object to retrieve the data frame with all the images and their respective ground truth.....	30
Figure 5.6: Here is where we import the pandas library so as to read the saved pickle file using google colab.....	30
Figure 5.7: We replace “Benign” and “Malignant” labels with 0 and 1 respectively, so as to feed these labels into the model that we build.....	30

Figure 5.8: This is the section where the image data along with the labels are split into test and training sets.....	31
<hr/>	
Figure 5.9: The training and testing images are removed from the data frame into a numpy array.....	31
Figure 5.10: Since we use a pre-trained GoogleNet model which is trained on RGB images, we convert our grayscale images to RGB images using the sckit learn library. Also, we reshape the image in the format that our model, which is coded using keras accepts data in.....	31
Figure 5.11: This is where we normalize the images by first changing their type to float and then dividing throughout by 255.....	32
Figure 5.12: We import all the libraries used to create the model. Keras has most of the functions that we need to build the model along with numpy for array manipulations....	32
Figure 5.13: This is the step where model architecture is laid out.....	32
Figure 5.14: Freezing the weights of the lower stack of layers.....	33
Figure 5.15: This step is where we set the type of optimizer we use, and also the loss function to be used is specified and then the entire model is compiled in the first ste....	33

## Chapter 1

# INTRODUCTION

### 1.1 What Is Breast Cancer?

Breast cancer starts when cells in the breast begin to grow out of control. These cells usually form a tumour that can often be seen on an x-ray or felt as a lump [1]. The tumour is malignant (cancer) if the cells can grow into (invade) surrounding tissues or spread (metastasize) to distant areas of the body. Breast cancer occurs almost entirely in women, but men can get breast cancer, too. Cells in nearly any part of the body can become cancer and can spread to other areas.

### 1.2 Where breast cancer starts?

Breast cancers can start from different parts of the breast. Most breast cancers begin in the ducts that carry milk to the nipple (ductal cancers). Some start in the glands that make breast milk (lobular cancers). There are also other types of breast cancer that are less common. A small number of cancers start in other tissues in the breast. These cancers are called sarcomas and lymphomas and are not really thought of as breast cancers.

Although many types of breast cancer can cause a lump in the breast, not all do. Many breast cancers are found on screening mammograms which can detect cancers at an earlier stage, often before they can be felt, and before symptoms develop. There are other symptoms of breast cancer you should watch for and report to a health care provider. It's also important to understand that most breast lumps are benign and not cancer (malignant). Non-cancerous breast tumours are abnormal growths, but they do not spread outside of the breast and they are not life threatening. But some benign breast lumps can increase a woman's risk of getting breast cancer. Any breast lumps or change needs to be checked by a health care professional to determine if it is benign or malignant (cancer) and if it might affect your future cancer risk

### **1.3 How Common Is Breast Cancer?**

Breast cancer is the most common cancer in American women, except for skin cancers. Currently, the average risk of a woman in the United States developing breast cancer sometime in her life is about 12%. This means there is a 1 in 8 chance she will develop breast cancer. This also means there is a 7 in 8 chance she will never have the disease.

### **1.4 Current year estimates for breast cancer**

The American Cancer Society's estimates for breast cancer in the United States for 2018 are:

- About 266,120 new cases of invasive breast cancer will be diagnosed in women.
- About 63,960 new cases of carcinoma in situ (CIS) will be diagnosed (CIS is non-invasive and is the earliest form of breast cancer).
- About 40,920 women will die from breast cancer.

### **1.5 Trends in breast cancer incidence**

In recent years, incidence rates have been the stable in white women and increasing slightly (by 0.3% per year) African American women. Breast cancer is more common in these women, compared to women of other races/ethnicities.

### **1.6 Trends in breast cancer deaths**

Breast cancer is the second leading cause of cancer death in women (only lung cancer kills more women each year). The chance that a woman will die from breast cancer is about 1 in 38 (about 2.6%). Death rates from female breast cancer dropped 39% from 1989 to 2015. Since 2007, breast cancer death rates have been steady in women younger than 50, but have continued to decrease in older women. These decreases are believed to be the result of finding breast cancer earlier through screening and increased awareness, as well as better treatments.

## 1.7 Early Detection of Breast Cancer

Finding breast cancer early and getting state-of-the-art cancer treatment are the most important strategies to prevent deaths from breast cancer. Breast cancer that's found early, when it's small and has not spread, is easier to treat successfully. Getting regular screening tests is the most reliable way to find breast cancer early. The American Cancer Society has screening guidelines for women at average risk of breast cancer, and for those at high risk for breast cancer.

## 1.8 Mammograms

Regular mammograms can help find breast cancer at an early stage, when treatment is most successful. A mammogram can find breast changes that could be cancer years before physical symptoms develop. Results from many decades of research clearly show that women who have regular mammograms are more likely to have breast cancer found early, are less likely to need aggressive treatment like surgery to remove the breast (mastectomy) and chemotherapy, and are more likely to be cured.

**Mammograms are not perfect. They miss some cancers.** And sometimes a woman will need more tests to find out if something found on a mammogram is or is not cancer. There's also a small possibility of being diagnosed with a cancer that never would have caused any problems had it not been found during screening. It's important that women getting mammograms know what to expect and understand the benefits and limitations of screening.



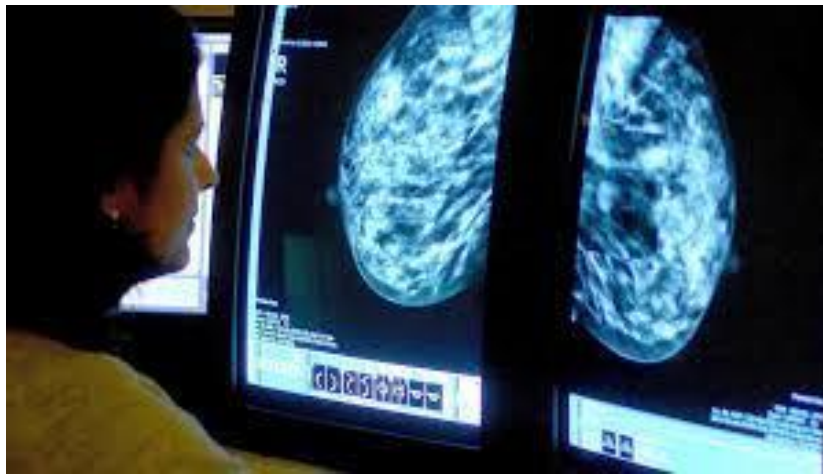


Fig 1.1: Mammogram [1]

## 1.9 Why breast screening is important?

A screening mammogram is the best method to detect breast cancer early for women over the age of 50 [2].



Fig 1.2: Breast Cancer in Woman [2]

1 in 8 women will develop breast cancer in their lifetime, and the risk of developing breast cancer increases with age.

## 1.10 Breast cancer detection technologies

### 1.10.1 Digital Mammography

In an attempt to improve x-ray mammography, several companies have developed digital mammography devices [3]. Unlike film mammography devices that produce an x-ray

image of the breast directly on photographic film, digital mammography devices (which still require breast compression) capture the x-ray image digitally (Figure 1.3). An array of detectors creates a digitized image that can be viewed and manipulated on a computer screen. In theory, this could enable better detection of tumours obscured by the dense breast tissue frequently seen in younger women. The ability to enlarge or adjust the contrast of questionable areas without requiring new x-ray exposure may facilitate the detection of lesions that have been missed by film mammography. The technology could also improve screening mammography by allowing electronic storage, retrieval, and transmission of mammograms. However, one important limitation of digital mammograms is that the images are not as finely detailed as film mammograms.

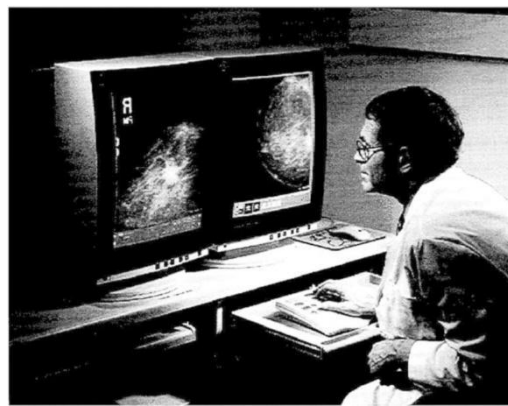


Fig 1.3: Examples of Full-Field Digital Mammography of the breast. SOURCE: General Electric Medical Systems [3]

Although digital mammography has been promoted as a major technical improvement over conventional mammography, preliminary studies have not yet confirmed a significant improvement in the accurate detection of breast cancers. More studies need to be done to assess its accuracy. One digital mammography machine has been approved by the FDA based on a small study of accuracy. Several other digital mammography units await FDA review and approval. To date, no studies have shown that digital mammography is more accurate or effective in reducing breast cancer deaths than film mammography.

### 1.10.2 Computer-Aided Detection (CAD)

Digital mammograms also make the use of computer-aided detection (CAD) systems easier. These systems use sophisticated computer programs to recognize patterns in images that might suggest a malignancy. If such patterns are detected, the CAD system notifies the radiologist, who can then examine the suspicious area more carefully. CAD can be used directly on digital mammograms or on conventional mammogram films that have been converted to a digital format.

Several studies suggest CAD can improve a radiologist's ability to detect and classify breast abnormalities on mammograms. One study suggested that CAD could have diminished the number of breast cancers missed in film mammography screening by nearly three-quarters. Other studies indicate that the addition of CAD to mammogram screening does not significantly boost the number of abnormalities inaccurately identified as potential tumours (false positives). More extensive studies must be done, however, to ensure that CAD does not lead to more false positive or negative results, and to define more clearly the value and appropriate use of this technology. The FDA recently approved one CAD detection system for breast cancer screening.

### 1.10.3 Ultrasound Imaging

X-ray mammograms are frequently followed by ultrasound imaging (Figure 1.4) to determine whether a mass that appeared on a mammogram is solid tissue or a harmless cyst containing fluid. Ultrasound imaging devices emit high-frequency sound waves, which penetrate the body. When these waves bounce off the boundaries between tissues in the body, they generate distinctive echoes that a computer uses to generate an image known as a sonogram. Because a fluid-filled cyst has a different “sound signature” than a solid mass, radiologists can reliably use ultrasound to detect cysts, which are commonly found in breasts.

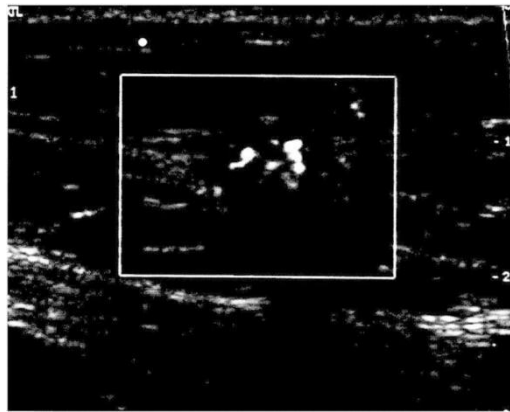


Fig 1.4: An example of an ultrasound image of the breast. SOURCE: Janet Baum, Director, Breast Imaging, Beth Israel Deaconess Medical Center, Boston, Massachusetts. [3]

Ultrasound imaging of the breast may also help radiologists evaluate some lumps that can be felt (palpable lesions) but are difficult to see on a mammogram, especially in women with dense breasts. One study of women with palpable lesions suggested that ultrasound was very accurate at diagnosing non-malignant abnormalities and could have eliminated the need for more than half the biopsies that were done. Other studies suggest that ultrasound may also be able to characterize non-palpable solid lesions as benign or malignant. Additional research suggests that ultrasound combined with x-ray mammography might improve the accuracy of breast cancer screening and also enable the detection of early-stage tumors in women with dense breasts. Further study is needed to assess the usefulness of ultrasound as a screening method used along with mammography.

Although ultrasound may be useful as an addition to mammography, it has limitations for breast cancer detection when used alone. Ultrasound often cannot detect small tumors (less than 5 mm or about one-quarter inch) and abnormalities (microcalcifications) linked to certain types of breast cancers. Recent improvements in ultrasound technology have the potential to overcome some of these limitations and to expand its usefulness in breast cancer detection. But their ultimate usefulness in breast cancer detection cannot be predicted at this stage of development.

#### 1.10.4 Magnetic Resonance Imaging (MRI)

Physicians have been using MRI for a wide variety of medical applications since it was FDA-approved for body imaging in 1985. MRI, generally considered to be a safe procedure, generates an image by measuring the responses of tissue components to a

magnetic field. Specialized MRI systems, designed for breast imaging (Figure 1.5) and approved by the FDA, show promise as a detection method to be used with mammography, especially for dense breasts

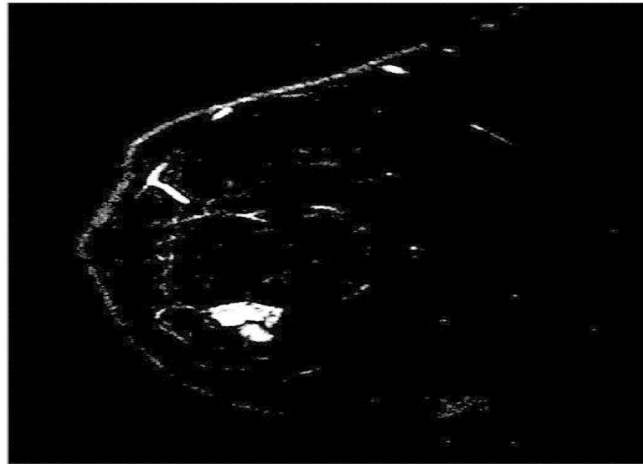


Fig 1.5: Example of a magnetic resonance image of the breast. SOURCE: Drs. D. Plewes and R. Shumak of Sunnybrook and Women's College Health Centre, University of Toronto.

Studies suggest that although MRI is highly sensitive at detecting tissue abnormalities that indicate cancer, it is sometimes unable to distinguish malignancies from other harmless tissue abnormalities in the breast. Also, like ultrasound, it cannot detect microcalcifications. Despite this limitation, an MRI could potentially detect the presence of a breast cancer in a patient whose mammogram, sonogram, and physical exam are not definitive.

Another possible use for MRI is to detect recurrent breast cancer in breasts previously subjected to lumpectomies, because unlike mammography, MRIs are usually not limited by scarring that can occur after surgery. Also, MRI can detect tumors in women with breast implants or dense breasts, both of which can interfere with interpretation of x-ray mammograms. Consequently, MRI may prove useful in the screening of high-risk young women (based on genetic testing or strong family history), who tend to have dense breasts. Preliminary results are encouraging in this regard, but further studies are needed to define the usefulness of MRI breast cancer screening in this population.

## Chapter 2

# TERMINOLOGIES

## 2.1 Convolutional neural network (ConvNet)

Convolutional Neural Networks are very similar to ordinary Neural Networks: they are made up of neurons that have learnable weights and biases [8]. Each neuron receives some inputs, performs a dot product and optionally follows it with a non-linearity. The whole network still expresses a single differentiable score function: from the raw image pixels on one end to class scores at the other. And they still have a loss function (e.g. SVM/Softmax) on the last (fully-connected) layer. ConvNet architectures make the explicit assumption that the inputs are images, which allows us to

encode certain properties into the architecture. These then make the forward function more efficient to implement and vastly reduce the amount of parameters in the network.

There are three main types of layers to build ConvNet architectures: **Convolutional Layer**, **Pooling Layer**, and **Fully-Connected Layer** (exactly as seen in regular Neural Networks). These layers are stacked to form a full ConvNet architecture.

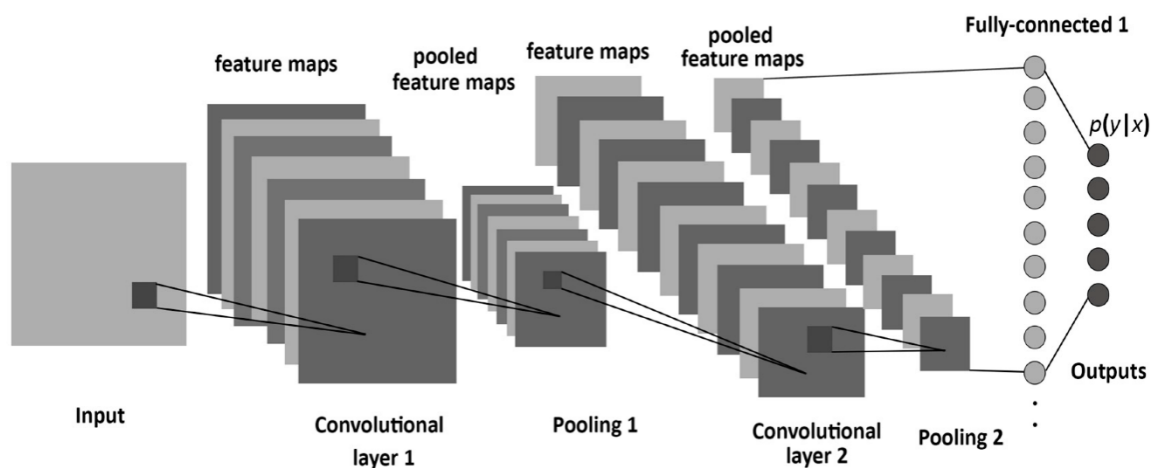


Fig 2.1 Convolutional neural network [8]

## 2.2 Dilation

Dilation is one of the two basic operators in the area of mathematical morphology, the other being erosion. It is typically applied to binary images, but there are versions that work on grayscale images[9].

The basic effect of the operator on a binary image is to gradually **enlarge the boundaries of regions** of foreground pixels (*i.e.* white pixels, typically). Thus areas of foreground pixels grow in size while holes within those regions become smaller.

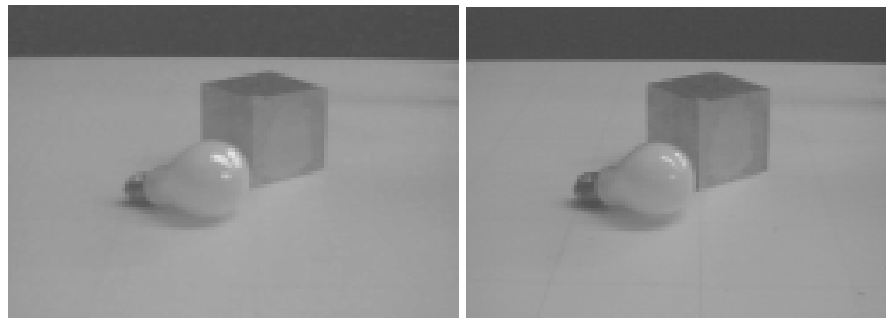


Fig 2.2: Original Image [9]

Fig 2.3: After Dilation [9]

## 2.3 Image Scaling

A 16-bit images that used in the dataset consists of 65536 possible graylevels. However, due to experimental conditions, much of the important image information may be contained within a narrow range towards the lower end of the grayscale. This is particularly the case for low light level experiments. Additionally, most computer monitors can only display 256 gray levels. Hence scaling allows you to select a range of gray values in a 16-bit image which can then either be displayed in the same image window or copied as a new 8-bit image. This will effectively rescale the intensities in the original image, allowing you more easily to see differences of grayscale values that might otherwise be impossible to discriminate visually. This technique does not affect your image data. It only affects the display of the data on your monitor.

## 2.4 Normalization

There are some variations on how to normalize the images but most seem to use these two methods:

1. Subtract the mean per channel calculated over all images
2. Subtract by pixel/channel calculated over all images

The natural approach would in my mind to normalize each image. An image taken in broad daylight will cause more neurons to fire than a night-time image and while it may



inform us of the time we usually care about more interesting features present in the edges etc.

## 2.5 Image Resizing

Image interpolation occurs when you resize or distort your image from one-pixel grid to another. Image resizing is necessary when you need to increase or decrease the total number of pixels, whereas remapping can occur when you are correcting for lens distortion or rotating an image. Zooming refers to increase the quantity of pixels, so that when you zoom an image, you will see more detail.

Interpolation works by using known data to estimate values at unknown points. Image interpolation works in two directions, and tries to achieve a best approximation of a pixel's intensity based on the values at surrounding pixels. Common interpolation algorithms can be grouped into two categories: adaptive and non-adaptive. Adaptive methods change depending on what they are interpolating, whereas non-adaptive methods treat all pixels equally.

Many compact digital cameras can perform both an optical and a digital zoom. A camera performs an optical zoom by moving the zoom lens so that it increases the magnification of light. However, a digital zoom degrades quality by simply interpolating the image. Even though the photo with digital zoom contains the same number of pixels, the detail is clearly far less than with optical zoom.

## Chapter 3

# METHODOLOGIES

## 3.1 Dataset Used

Digital Database for Screening Mammography (DDSM) [5] is used for the experiments. The dataset composes of nearly 2620 samples of mammograms which are grey level images. Each mammogram has additional information on the segmentation of micro-calcifications regions which are marked manually by the radiologist [6].

Micro-calcifications are calcium deposit in breast tissues that can potentially cause cancer. The dataset consists of four types of cases labelled as normal, benign without call-back, benign and malignant. But only benign and malignant classes are used. Presence of malignant tumour would mean the patient has a cancerous tumour and benign tumour would mean not cancerous tumour. Some mammograms are discarded since they have low resolution caused by the segmentation operation of the dataset that significantly affect the classification; therefore, 1755 images remaining from 1784 images are used in the experiments. The first 1405 of the 1755 images are used for training and the remaining 350 images are used as test data. 705 of the 1405 images and 168 of 350 images are of malignant class

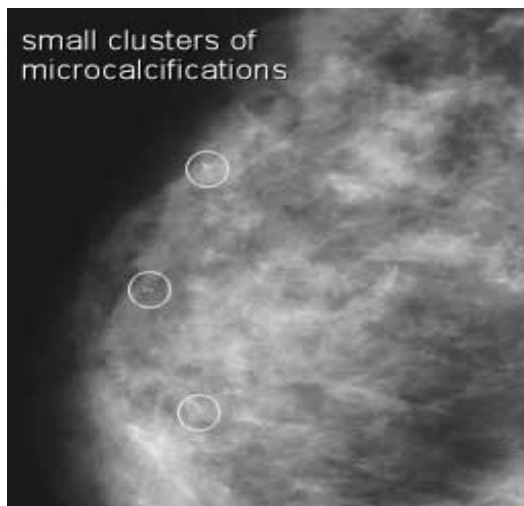


Fig 3.1 Mammogram image [10]

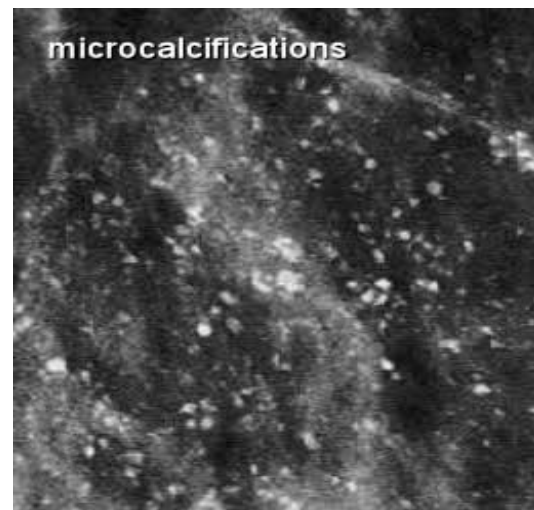


Fig 3.2 : Microcalcifications zoomed [10]

### 3.2 Data Preprocessing

Since the microcalcifications are small relative to the breast masses, signal-to-noise ratio of the dataset is improved for effective classification. First, each pixel is scaled down from being a 16-bit pixel to an 8-bit pixel. This is necessary because we don't require very huge pixel values for our computation as it would slow down processing unnecessarily. Each image's contrast is enhanced scaling each pixel by  $255/(\text{maximum})$

intensity) and the others accordingly. Then, to boost the microcalcification signal, dilation is performed.

During training, mean pixel value of the dataset is subtracted from each image. Then the image is resized to 224 pixels

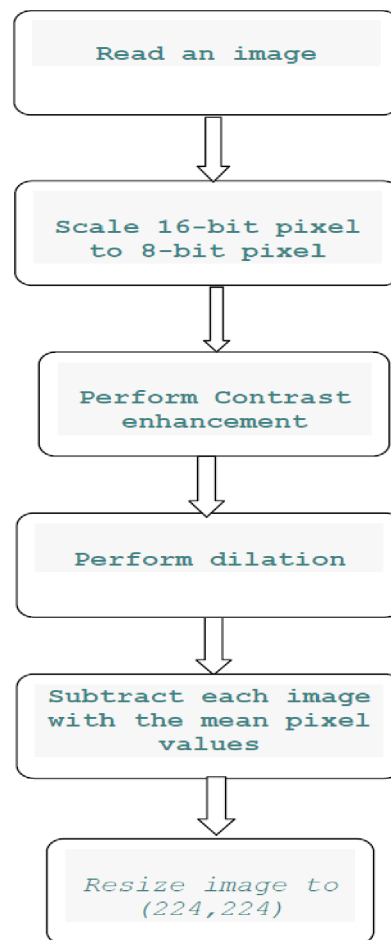


Fig 3.3: Image Processing Implementation

### 3.3 Data Augmentation

Training set without data augmentation is prone to overfitting since the size of the data is not sufficient to make eloquent generalization. In order to increase the data size, online and offline data augmentation techniques are used. Before training, each image is rotated by  $180^\circ$  and if the image is square, the image is also rotated by  $90^\circ$  and

270°. Furthermore, random Gaussian noise is added to each image. Hence, the dataset is augmented to 5324 images containing 2937 malignant samples. During training, each image's mirror is taken and random 224x224 crops are obtained to reduce overfitting.



Fig 3.4: Original image



Fig 3.5: Augmented image (180° rotate over the original)

### 3.4 Network Architecture

The convolutional neural network architecture selected for the classification is GoogLeNet [4], which uses its authentic inception modules to overcome the vanishing gradient problem [4]. For experiments, a pre-trained model on the ImageNet dataset [7] is used except the last fully-connected layer's hidden units are reduced to 2 from 1000, since in this study there are only two classes namely benign and malignant. Model's learning rate is kept at  $\eta = 10^{-4}$  with Adam optimizer.

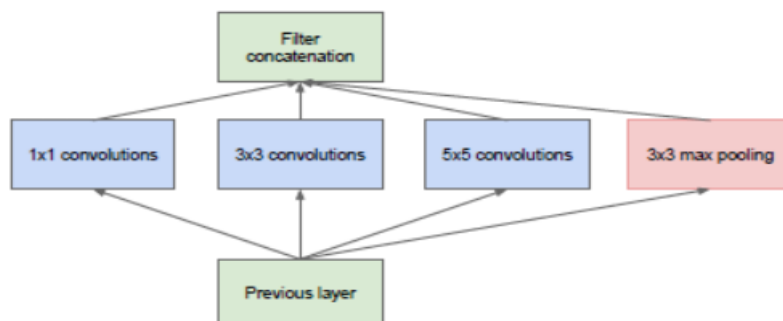


Fig 3.6: Inception model (naïve version) [11]

The figure above shows a sub-module of the inception model. The inspiration comes from the idea that you need to make a decision as to what type of convolution you want to make at each layer: The core essence of the inception model lies in the fact that it uses the model to decide whether to use a 3x3 filter, 5x5 filter, or a 7x7 filter. You do this by doing each convolution in parallel and concatenating the resulting feature maps before going to the next layer. Here lies the intuition of the inception model.

## Chapter 4

# SOFTWARE TOOLS

## 4.1 Keras

Keras is an open source machine learning library. It is a high-level neural networks API, written in Python and capable of running on top of TensorFlow (another widely used neural networks library). It was developed with a focus on enabling fast experimentation and allows for easy and fast prototyping (through user friendliness, modularity, and

extensibility). It also supports both convolutional networks and recurrent networks, as well as combinations of the two. It also runs seamlessly on CPUs and GPUs.

We also used keras to load our Inception model which was pretrained using the ImageNet database.

## 4.2 Pandas

**Pandas** is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. In particular, it offers data structures and operations for manipulating numerical tables and time series. The CBIS-DDSM database provided a comma separated values (CSV) file that had a record of each DICOM image file, its path in the image folder, and other information like their labels, abnormality type, left or right breast, etc. Importing the CSV file required the package **Pandas** to load the file as a data frame object. This *data frame* object could then be easily manipulated to retrieve the images present in the image folder, and to also split the data into training and testing slots. The labels of the data could also be retrieved from the dataframe.

```
In [17]: import pandas as pd

# Loading one such CSV file into df
df= pd.read_csv('calc_case_description_test_set.csv', index_col=False)
df.head()
```

Out[17]:

	patient_id	breast density	left or right breast	image view	abnormality id	abnormality type	calc type	calc distribution	assessment	pathology
0	P_00038	2	LEFT	CC	1	calcification	PUNCTATE- PLEOMORPHIC	CLUSTERED	4	BENIGN
1	P_00038	2	LEFT	MLO	1	calcification	PUNCTATE- PLEOMORPHIC	CLUSTERED	4	BENIGN
2	P_00038	2	RIGHT	CC	1	calcification	VASCULAR	NaN	2	BENIGN_WITHOUT_CALLBACK
3	P_00038	2	RIGHT	CC	2	calcification	VASCULAR	NaN	2	BENIGN_WITHOUT_CALLBACK
4	P_00038	2	RIGHT	MLO	1	calcification	VASCULAR	NaN	2	BENIGN_WITHOUT_CALLBACK

Fig 4.1: Loading the CSV file as a Data frame object and printing the first 5 rows.

### 4.3 OpenCV

OpenCV is a library of programming functions mainly aimed at real-time computer vision. It contains a plethora of image processing tools that we selectively operate on the Region of Interest (ROI) images. For our project we use operations like **dilation**, **contrast stretching** and **resizing** to modify and enhance the image dataset, before we train our model for prediction.

### 4.4 Numpy

NumPy is another library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. Or in this case, an array of pixels.

We use NumPy to operate on these images and modify the pixel data, along with OpenCV to perform image processing on the images.

```
In [21]: # Reading the image
path=newcsv['ROI mask file path'][8]
img=pdicom.read_file(path,force=True)
imgpix=img.pixel_array

# Scaling 16-bit pixel to 8-bits
imgpix=((imgpix/65535)*255).astype(np.uint8)

# Resizing image to (224,224)
imgpix=cv2.resize(imgpix,(224,224))

#Contrast enhancement
equ = cv2.equalizeHist(imgpix)

#dilation
k1 = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (4, 5))
k4 = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 7))
dil= cv2.dilate(equ,k1,iterations = 1)

# Stacking all the images together for comparison
res = np.hstack((imgpix,equ,dil)) #stacking images side-by-side
plt.imshow(res)
```

Out[21]: <matplotlib.image.AxesImage at 0x119b964e0>

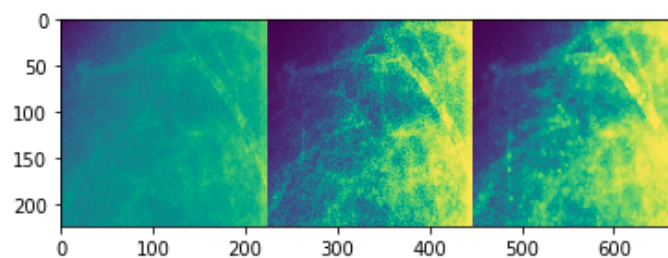


Fig 4.2: Utilizing the tools of both OpenCV and Numpy to display the original image(left), equalized image(middle) and final dilated image(right).

## 4.5 Jupyter Notebook

Notebook documents (or “notebooks”, all lower case) are documents produced by the **Jupyter Notebook App**, which contain both computer code and rich text elements (paragraph, equations, figures, links, etc). The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc.) as well as executable

These notebook documents were used to split our project’s tasks into manageable chunks, spanning across the notebook, which we would edit and rerun according to our needs. It



simplified the job of gathering our code under one container and allowed us to analyze and explore our image dataset via Matplotlib's plot visualizations.

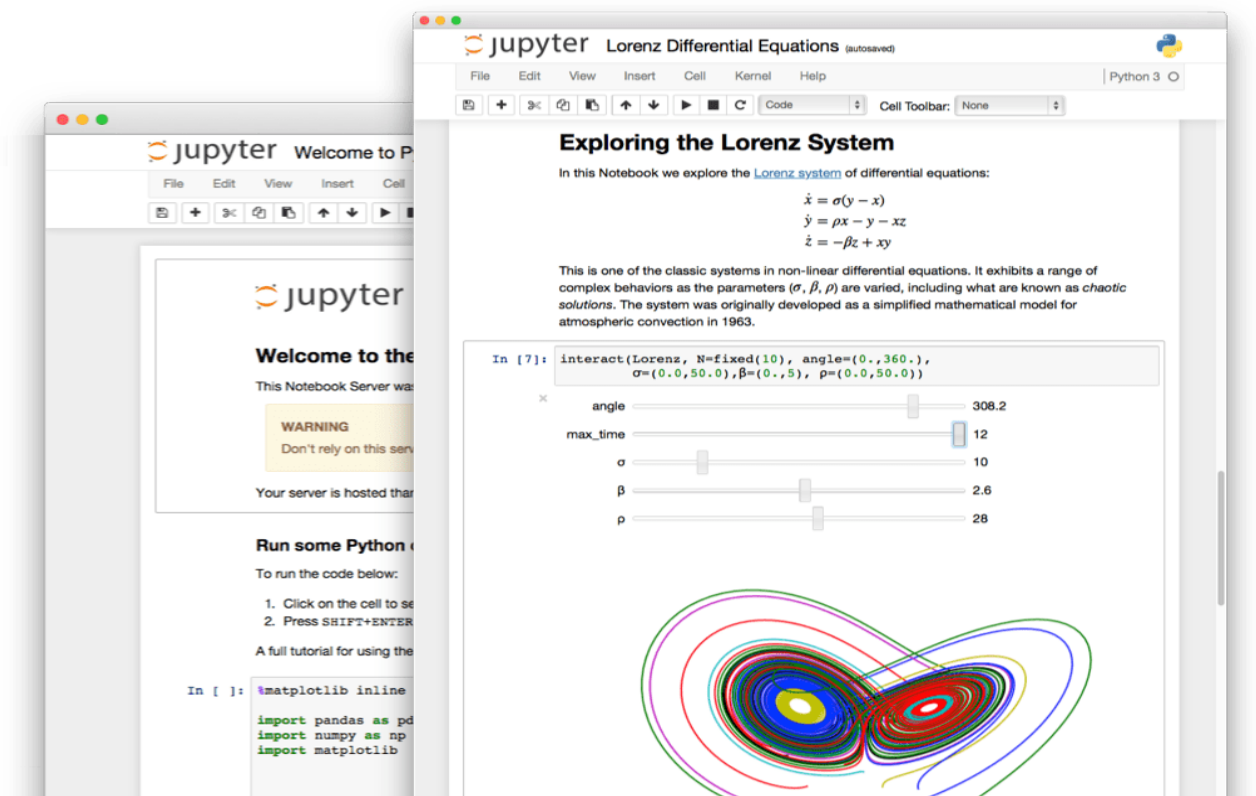


Fig 4.3: Notebooks which can be run to perform data analysis

The visualizations took the form of grayscale images displayed inline within the document, and a side-by-side comparison of these images before and after image preprocessing could also be viewed.

## 4.6 Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy mentioned earlier. We employ the library to view and display images inline within the Jupyter notebook.

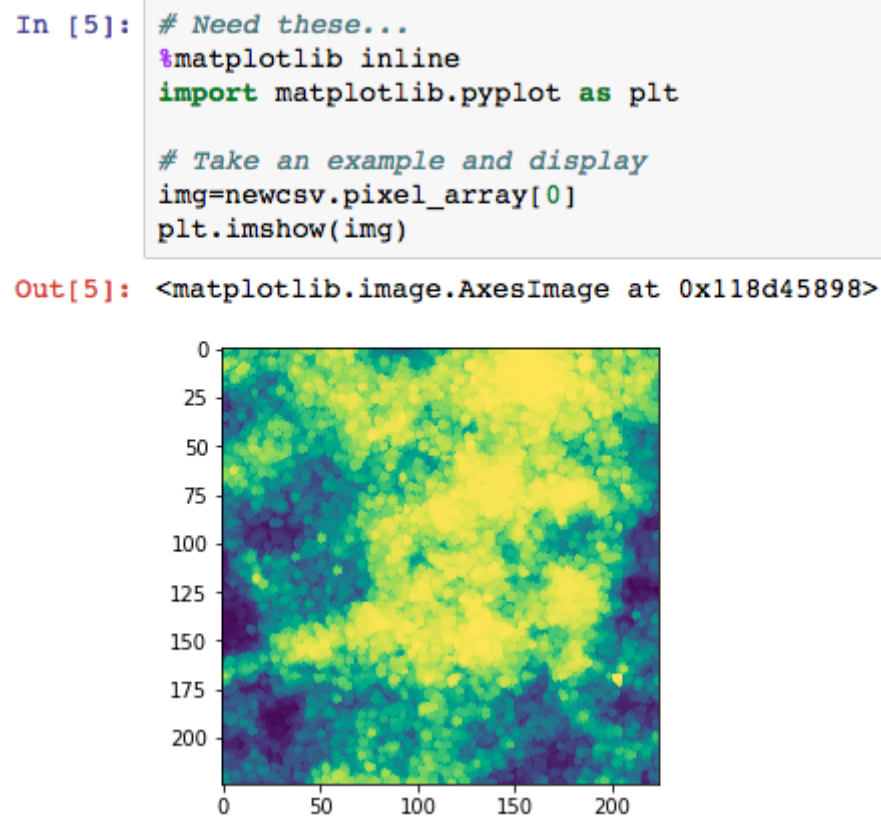


Fig 4.4: Using Matplotlib to plot the region of interest image

## 4.7 PyDICOM

Digital Imaging and Communications in Medicine (**DICOM**) is a standard for storing and transmitting medical images enabling the integration of medical imaging devices such as scanners, servers, workstations, printers, network hardware, and picture archiving and communication systems (PACS) from multiple manufacturers.

**pydicom** is a python package for working with DICOM files. It was made for inspecting and modifying DICOM data in an easy "pythonic" way. The modifications can be written again to a new file. We use the package to read and display mammogram images in our dataset. The following information can be retrieved from a DICOM file and its image is can also be displayed.

```
In [8]: import dicom as pdicom
path=newcsv['ROI mask file path'][0]
img=pdicom.read_file(path,force=True)

imgpix=img.pixel_array
plt.imshow(imgpix)
img
```

```
Out[8]: (0008, 0005) Specific Character Set          CS: 'ISO_IR 100'
(0008, 0016) SOP Class UID                        UI: Secondary Capture Image Storage
(0008, 0018) SOP Instance UID                    UI: 1.3.6.1.4.1.9590.100.1.2.240727997110773773805535351310689830335
(0008, 0020) Study Date                          DA: '20170829'
(0008, 0023) Content Date                       DA: '20160503'
(0008, 0030) Study Time                         TM: '180052'
(0008, 0033) Content Time                      TM: '105948.640000'
(0008, 0050) Accession Number                  SH: ''
(0008, 0060) Modality                          CS: 'MG'
(0008, 0064) Conversion Type                   CS: 'WSD'
(0008, 0090) Referring Physician's Name       PN: ''
(0008, 103e) Series Description                 LO: 'cropped images'
(0010, 0010) Patient's Name                    PN: 'Calc-Test_P_00038_LEFT_CC_1'
(0010, 0020) Patient ID                       LO: 'Calc-Test_P_00038_LEFT_CC_1'
(0010, 0030) Patient's Birth Date             DA: ''
(0010, 0040) Patient's Sex                    CS: ''
(0013, 0010) Private Creator                  OB: b'CTP '
(0013, 1010) Private tag data                 OB: b'CBIS-DDSM '
(0013, 1013) Private tag data                 OB: b'43372602'
(0018, 0015) Body Part Examined               CS: 'BREAST'
(0018, 1016) Secondary Capture Device Manufactur LO: 'MathWorks'
(0018, 1018) Secondary Capture Device Manufactur LO: 'MATLAB'
(0020, 000d) Study Instance UID                UI: 1.3.6.1.4.1.9590.100.1.2.161465562211359959230647609981488894942
(0020, 000e) Series Instance UID              UI: 1.3.6.1.4.1.9590.100.1.2.419081637812053404913157930753972718515
(0020, 0010) Study ID                         SH: 'DDSM'
(0020, 0011) Series Number                    IS: '1'
(0020, 0013) Instance Number                  IS: '1'
(0020, 0020) Patient Orientation              CS: 'CC'
```

Fig 4.5: First set of Data Attributes related to the mammogram can be seen above

```
<matplotlib.image.AxesImage at 0x11935bcc0>
```

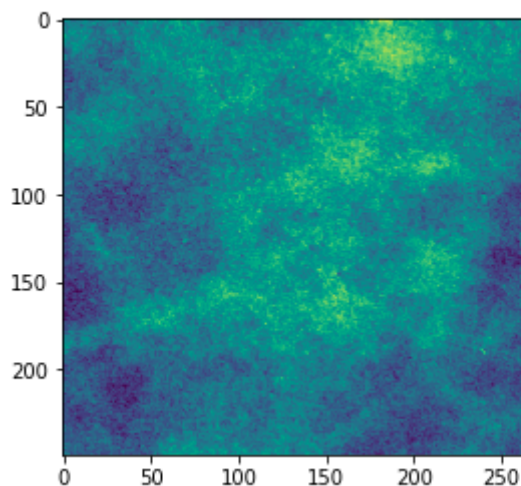


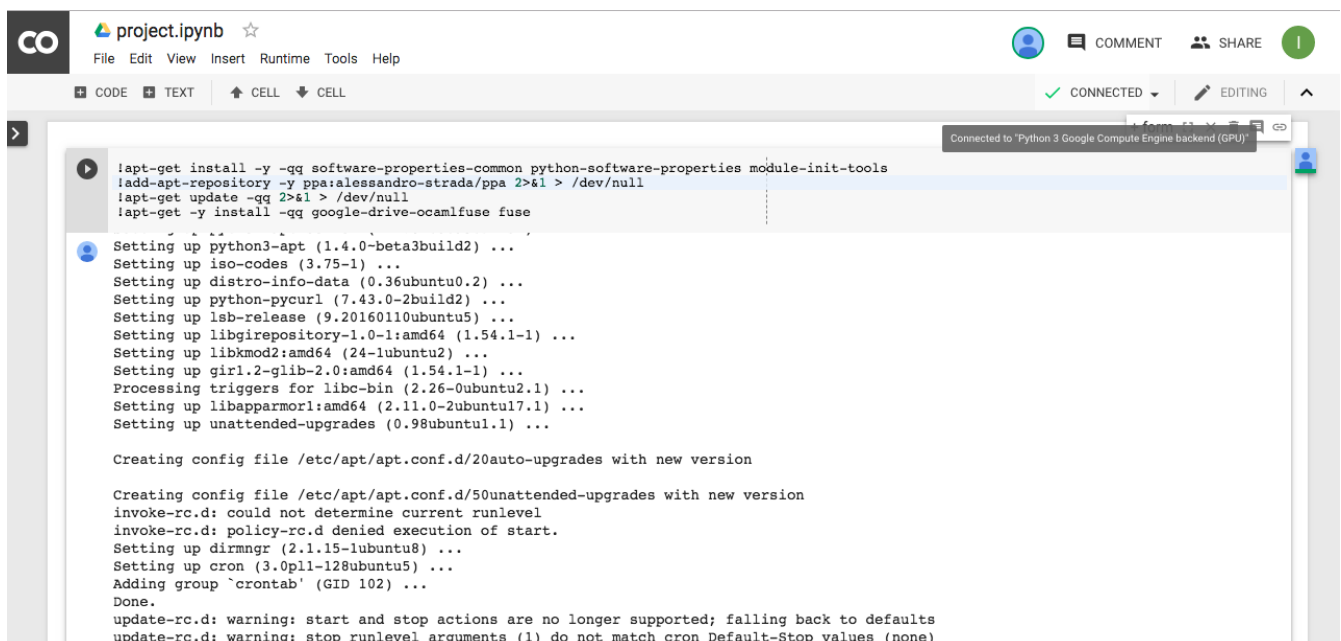
Fig 4.6: Mammogram Image (ROI)

Manipulating the image's pixel data requires packages like Numpy and packages like Matplotlib to display it.

## 4.8 Google's Colaboratory

Colaboratory is a Google research project created to help disseminate machine learning education and research. It's a **Jupyter notebook environment** that requires no setup to use and runs entirely in the cloud.

We were in need of a GPU to train our model and test it in and Colaboratory gave us this setup along with notebook functionality and use similar to Jupyter Notebook App mentioned earlier.



```
project.ipynb
File Edit View Insert Runtime Tools Help
CODE TEXT CELL CELL
CONNECTED EDITING
Connected to 'Python 3 Google Compute Engine backend (GPU)'

!apt-get install -y -qq software-properties-common python-software-properties module-init-tools
!add-apt-repository -y ppa:alessandro-strada/ppa 2>&1 > /dev/null
!apt-get update -qq 2>&1 > /dev/null
!apt-get -y install -qq google-drive-ocamlfuse fuse

Setting up python3-apt (1.4.0-beta3build2) ...
Setting up iso-codes (3.75-1) ...
Setting up distro-info-data (0.36ubuntu0.2) ...
Setting up python-pycurl (7.43.0-2build2) ...
Setting up lsb-release (9.20160110ubuntu5) ...
Setting up libgirepository-1.0-1:amd64 (1.54.1-1) ...
Setting up libxmod2:amd64 (24-lubuntu2) ...
Setting up gir1.2-glib-2.0:amd64 (1.54.1-1) ...
Processing triggers for libc-bin (2.26-0ubuntu2.1) ...
Setting up libapparmor1:amd64 (2.11.0-2ubuntu17.1) ...
Setting up unattended-upgrades (0.98ubuntu1.1) ...

Creating config file /etc/apt/apt.conf.d/20auto-upgrades with new version

Creating config file /etc/apt/apt.conf.d/50unattended-upgrades with new version
invoke-rc.d: could not determine current runlevel.
invoke-rc.d: policy-rc.d denied execution of start.
Setting up dirmngr (2.1.15-1ubuntu8) ...
Setting up cron (3.0pl1-128ubuntu5) ...
Adding group `cronstab' (GID 102) ...
Done.
update-rc.d: warning: start and stop actions are no longer supported; falling back to defaults
update-rc.d: warning: stop runlevel arguments (1) do not match cron Default-Stop values (none)
```

Fig 4.7: Google Colaboratory notebook

## Chapter 5

# SOFTWARE IMPLEMENTATION

Our first task is to elaborate our workflow, and since we used the Jupyter Notebook app and GoogleCOLAB notebooks (for our machine learning framework) to structure our code, we will attempt to do this using snapshots of the code.

One of the first steps in our project workflow was to perform image preprocessing on the entire dataset, along with its augmentation. So a set of these snapshots are shown below; with helpful comments to interpret them.

```
In [ ]: # Import these packages
import matplotlib inline
import matplotlib.pyplot as plt
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import dicom as pdicom #Read mammogram images stored in DICOM files
import cv2
```

Fig 5.1: Importing the necessary packages. Packages specified in the **Software Tools** section, detail what each module is used for.

```

In [ ]: ## Load the original csv as labels_df

pth=''
cs1= pd.read_csv(pth+'calc_case_description_test_set.csv',index_col=False)
cs2= pd.read_csv(pth+'calc_case_description_train_set.csv',index_col=False)
cs3=pd.concat([cs1,cs2])

# Removing cases of Benign Without Callback
cs3[cs3['pathology']!='BENIGN_WITHOUT_CALLBACK']

# There is a trailing line break at the end of each image's path which is removed here
cs3['ROI mask file path']=pth+'DOI/'+cs3['ROI mask file path']

# Create empty dataframe first
labels_df=pd.DataFrame()

# Check whether all images are present by looking them up
# and creating a new DataFrame object with only those image that are present
for x in range(cs3.shape[0]):
    row=cs3.iloc[x]
    imgloc=row['ROI mask file path']
    try:
        img=pdicom.read_file(imgloc,force=True)
        labels_df=labels_df.append(row,ignore_index=True)
    except:
        pass
print('# of Images present:',labels_df.shape[0])
print('# of Images missing:',cs3.shape[0]-labels_df.shape[0])
print('Total:',cs3.shape[0])

```

Fig 5.2: Loading the CSV file of the image database as a Pandas dataframe object and performing a check to see if all the images recorded in the file are present in the image folder. If not, they are ignored and a new CSV file is created, keeping track of only those images that are present.

```
In [ ]: # Define the preprocessing functions
from skimage.transform import rotate
def preprocess_image(path):
    img=pdicom.read_file(path,force=True)
    imgpix=img.pixel_array

    # Scaling 16-bit pixel to 8-bits
    imgpix=((imgpix/65535)*255).astype(np.uint8)

    # Contrast enhancement
    imgpix = cv2.equalizeHist(imgpix)

    #dilation
    k1 = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (4, 5))
    k4 = cv2.getStructuringElement(cv2.MORPH_ELLIPSE, (5, 7))
    imgpix= cv2.dilate(imgpix,k1,iterations = 1)

    # Subtracting each image with the mean pixel values
    imgpix = (imgpix-imgpix.mean()).astype(np.uint8)

    # Resizing image to (224,224)
    imgpix=cv2.resize(imgpix,(224,224))

    return imgpix
```

Fig 5.3: Defining the image preprocessing function is another crucial step in determining what tools we employ to “enhance” the image.

Down below you can see how steps like **Scaling**, **Contrast Enhancement** and **Dilation** are the image processing functions we use. This is followed by a subtraction of each image with the overall mean image of the dataset known as **Mean Subtraction**. It is finally resized to (224,224) pixel dimensions and returned to the calling function.

```
In [ ]: # Create new CSV with the images stored with it as newcsv and stored as new.pkl
newcsv=pd.DataFrame()
ROTATE_LEVELS=[0,90,180,270]
for x in range(labels_df.shape[0]):
    row=labels_df.iloc[x]
    imgloc=row['ROI mask file path']
    img=preprocess_image(imgloc)
    for y in ROTATE_LEVELS:
        if y:
            imgpix=rotate(img,y,preserve_range=True).astype(np.uint8)
        else:
            imgpix=img
    row['pixel_array']=imgpix
    newcsv=newcsv.append(row,ignore_index=True)

# Saving the dataframe as a pickle file
newcsv.to_pickle('new.pkl')
print("Done!")
```

Fig 5.4: Data augmentation and the calling of the *preprocess* function is done here.

Here the data augmentation involves rotating the image by 90, 180 and 270 degrees to expand our dataset to 4 times the original size. The new images generated along with their ground truth information is replicated in a new Pandas dataframe object. Here the images

itself are stored as a pixel array of size (224,224). This dataframe object file is finally saved as a *pickle* file named appropriately as *new.pkl*; A *pickle* file is a binary serialization of our dataframe object that can be loaded seamlessly into our notebook document.

```
In [4]: # Load the pickle file as a dataframe object
newcsv=pd.read_pickle('new.pkl')
```

Fig 5.5: This line of code (that can be added into any notebook), loads the pickle object to retrieve the data frame with all the images and their respective ground truth.

Saving the images in this way allowed us to load the images as a single file, and would also alleviate all the problems related to training and testing the neural network model.

From this point on, we implement our code on Google colab and the following snapshots are from the Google colab notebook.

```
import pandas as pd
newcsv=pd.read_pickle("newcsv.pkl") # read the csv
req_content=newcsv[['pathology',"pixel_array"]] # keep only the required content
```

Fig 5.6: Here is where we import the pandas library so as to read the saved pickle file using google colab.

The required content is then kept from the entire data frame like the pixel array and the pathology, i.e. if the patient had benign or malignant tumour.

```
[ ] d = {'BENIGN': 0, 'MALIGNANT': 1} # Map the labels to 0 and 1
req_content['pathology']=req_content['pathology'].map(d)
```

Fig 5.7: We replace “Benign” and “Malignant” labels with 0 and 1 respectively, so as to feed these labels into the model that we build.



```
[ ] res_m=req_content[req_content['pathology']== 1]      # Here we split the dataset equally
    res_b=req_content[req_content['pathology']==0]      #
                                                #
    train_m=res_m.sample(frac=0.85,random_state=200)   #
    train_b=res_b.sample(frac=0.85,random_state=200)

    test_m=res_m.drop(train_m.index)
    test_b=res_b.drop(train_b.index)
```

Fig 5.8: This is the section where the image data along with the labels are split into test and training sets.

Also the test set is made to have equal number of malignant and benign images

```
train_images=array((train["pixel_array"]).tolist())   # first convert the series to list
test_images=array(test["pixel_array"].tolist())        # and then convert it into a numpy array

train_labels=train["pathology"].tolist()              #
test_labels=test["pathology"].tolist()                #
```

Fig 5.9: The training and testing images are removed from the data frame into a numpy array.

The labels are just converted to lists and kept as it is.

```
import skimage.color as color
train_images=color.gray2rgb(train_images)              # convert from grayscale to color
test_images=color.gray2rgb(test_images)               # so as to feed it to the neural network

train_images = train_images.reshape(train_images.shape[0], 224, 224, 3) # reshape the array
test_images = test_images.reshape(test_images.shape[0], 224, 224, 3) # (this is the format in
                                                                    # which keras accepts data)
```

Fig 5.10: Since we use a pre-trained GoogleNet model which is trained on RGB images, we convert our grayscale images to RGB images using the scikit learn library. Also, we reshape the image in the format that our model, which is coded using keras accepts data in.

```
train_images =train_images.astype('float32')
test_images= test_images.astype('float32')

train_images/=255          # Pixel normalization
test_images/=255          #
```

Fig 5.11: This is where we normalize the images by first changing their type to float and then dividing throughout by 255.

This step is necessary since this step acts as a **feature normalization** step before being passed to the model.

```
import numpy as np
from keras import layers
from keras.layers import Input,Dense,BatchNormalization,Flatten,Dropout,GlobalAveragePooling2D
from keras.models import Model
from keras.utils import layer_utils
from keras.optimizers import Adam
from keras.callbacks import ModelCheckpoint
from keras.callbacks import EarlyStopping
import keras.backend as K
from keras.applications.inception_v3 import InceptionV3
```

Fig 5.12: We import all the libraries used to create the model. Keras has most of the functions that we need to build the model along with numpy for array manipulations.

```
base_model = InceptionV3(weights='imagenet', include_top=False) #load the inception model
#without the last FC layer
x = base_model.output
x = GlobalAveragePooling2D()(x)
x = Dense(128,activation='relu')(x)
predictions = Dense(1, activation='sigmoid')(x)
m = Model(inputs=base_model.input, outputs=predictions)
```

Fig 5.13: This is the step where model architecture is laid out.

First the inception model is used from the keras library without the last fully connected layer since the imagenet classifier was trained on 1000 labels and we need just two. Remaining steps stack layers over the existing model.

```
[ ] for layer in m.layers[:165]:  
    layer.trainable = False  
    for layer in m.layers[165:]:  
        layer.trainable = True
```

Fig 5.14: Freezing the weights of the lower stack of layers.

This is a crucial step. Since the model is pre-trained on images, the lower stack of layers detects basic features like curves, edges, colour variations, etc.

Hence we train our dataset by freezing the weights of the lower stack of layers so as to not change these rudimentary features required for any image classification problem.

```
m.compile(loss="binary_crossentropy", optimizer="adam", metrics=["accuracy"])  
model_info = m.fit(x=train_images, y=train_labels, batch_size=130, epochs=epochs,  
                  verbose=1, validation_data=(test_images, test_labels))
```

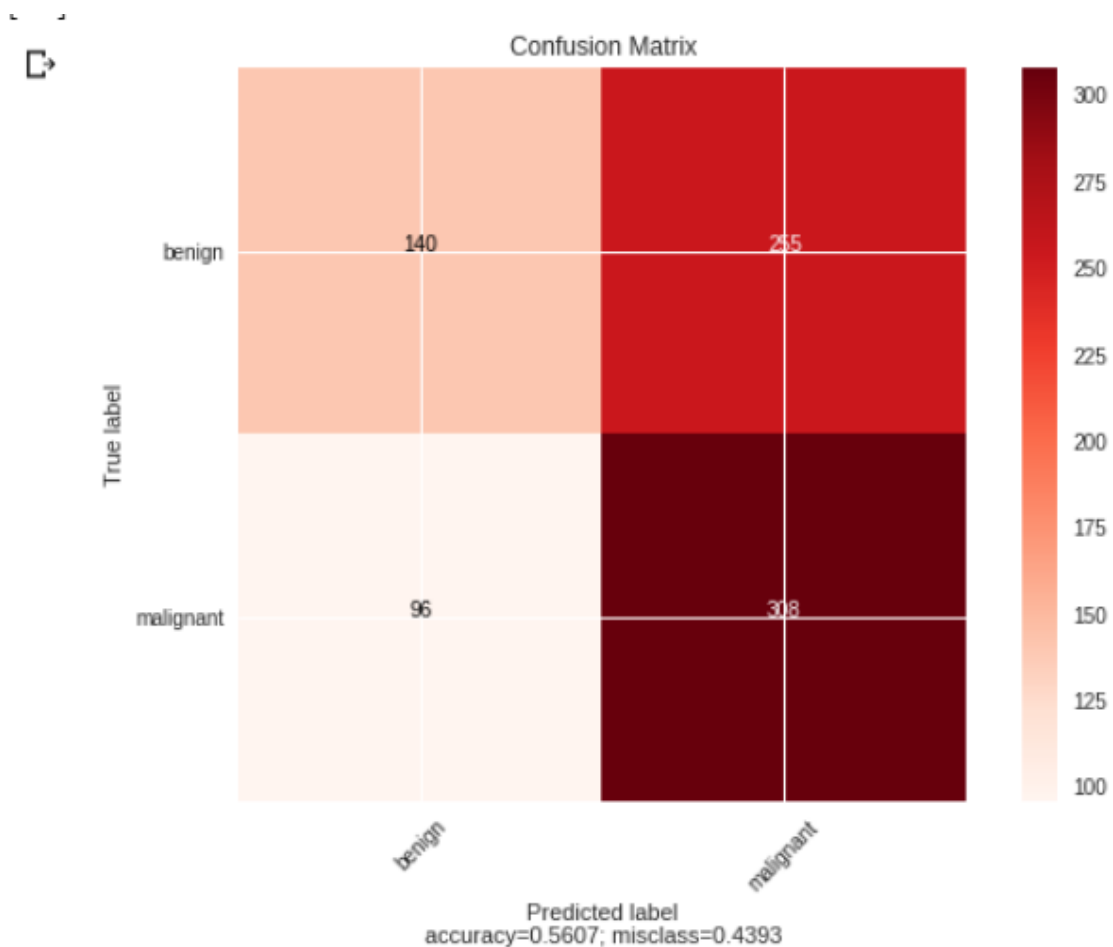
Fig 5.15: This step is where we set the type of optimizer we use, and also the loss function to be used is specified and then the entire model is compiled in the first step.

The next step is where the model is trained and validated on our dataset. This step takes very long time due to the large data that it acts on

## EXPERIMENTAL RESULTS

The network was trained and tested using the parameters like accuracy, precision, recall and F1 scores. Accuracy is defined as the number of correct predictions divided by the total number of predictions. Precision is defined as the number of true positives divided by the sum of true and false positives. Here, positive is considered to be malignant case. Recall is defined as the number of true positives divided by the sum of true positives and false negatives. F1 score is a harmonic mean of precision and recall.

We obtain an accuracy of 56%, precision of 53%, recall of 76%, F1 score of 62.4%, The confusion matrix gives a better picture into the result.



## CONCLUSION

We infer from the results that a high recall rate of 76% is desirable since when screening for cancer, it is crucial that patients having cancer are rightly detected to have cancer. Other metrics can be improved significantly with more data, since we were dealing with a limited dataset. Overall we observe that the loss function kept decreasing during training, which means that the model could learn more but didn't have enough data. In order to avoid overfitting, we restricted the number of iterations around our dataset to 5.

## **FUTURE SCOPE**

Our model achieved an accuracy of 56%, with a limited dataset and showcases the strengths and abilities of Deep Learning architectures in classification tasks. It can be said that our model predicts its classes well; given our confusion matrix. Our hope is to see that model be incorporated and used within the medical community as an aid to radiologists, after they have isolated the cancerous cells in the breast tissue. Major changes and improvements in deep learning & image classification models in the coming years, will continue to serve the medical community towards this endeavour.

## REFERENCES

- [1] Mammogram, <https://www.nbcnews.com/feature/30-seconds-to-know/video/when-should-you-get-a-mammogram-543493699715>
- [2] Breast cancer statistics, <https://www.ourbodiesourselves.org/2017/04/breast-cancer-breast-implants-making-decisions-moving-forward/>
- [3] Breast cancer diagnosis technologies, <https://www.ncbi.nlm.nih.gov/books/NBK223396/>
- [4] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 1-9).
- [5] Heath, M., Bowyer, K., Kopans, D., Moore, R., & Kegelmeyer, W. P. (2000, June). The digital database for screening mammography. In Proceedings of the 5th international workshop on digital mammography (pp. 212-218). Medical Physics Publishing.
- [6] Heath, M., Bowyer, K., Kopans, D., Kegelmeyer Jr, P., Moore, R., Chang, K., & Munishkumaran, S. (1998). Current status of the digital database for screening mammography. In Digital mammography (pp.457-460). Springer Netherlands.
- [7] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S. & Berg, A. C. (2015). Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3), (pp.211-252).
- [8] Convolutional neural network, from <http://cs231n.github.io/convolutional-networks/>
- [9] Dilation, from <https://homepages.inf.ed.ac.uk/rbf/HIPR2/dilate.htm>

[10] Ensemble of Convolutional Neural Networks for Classification of Breast Micro calcification from Mammograms, Egemen Sert, Seyda Ertekin, Ugur Halici, Senior Member, IEEE (pp 1-4)

[11] Inception model-naïve version, from

<https://hacktilldawn.com/2016/09/25/inception-modules-explained-and-implemented/>