

Software Engineering Department

Braude College

Capstone Project Phase B

**Breast Cancer Classification using CNN**

**23-1-R-1**

**Supervisor:**

 Ronen Zilber, [rzilber1@gmail.com](mailto:rzilber1@gmail.com)‏

**Students:**

Shenhav Hezi, [Shenhav.Hezi@e.braude.ac.il](mailto:Shenhav.Hezi@e.braude.ac.il)

Tomer Boyanjo, [Tomer.Boyanjo@e.braude.ac.il](mailto:Tomer.Boyanjo@e.braude.ac.il)

Github Link: <https://github.com/Shenhav26/Final_project>

**Table of content**

1. Introduction 3

2. Background and Related Work 4

2.1 ANN 4

2.1.1 Weights 4

2.1.2 Activation function 4

 2.2 CNN 5

2.2.1 Convolutional layers 5

2.2.2 Pooling layers 5

2.2.3 Fully connected layer 6

3. Project review and process description 7

3.1 Architecture – DenseNet121 7

3.1.1 Vanishing Gradient problem solve 7

3.1.2 DenseNet Components 8

3.1.2.1 Connectivity 8

3.1.2.2 DenseBlocks 8

3.1.2.3 Growth Rate 9

3.1.2.4 Bottleneck layers 9

3.1.3 DenseNet Architecture 9

3.1.4 Advantages of the DenseNet 10

3.2 Architecture – Inception 11

3.3 Research process 12

3.4 Results 13

3.4.1 Mammography 14

3.4.2 Histopathology 16

3.4.3 DenseNet 121 VS Inception V3 18

3.4.4 Confusion matrix 19

3.4.5 Discussions and conclusions 20

4. Product 21

4.1 Sequence diagram 21

4.2 Flowchart diagram 21

4.3 GUI 22

4.4 Description of environment 23

5. References 24

**Abstract**

Cancer is a group of diseases involving abnormal cell growth According to American cancer society, Cancer continues to be the second most common cause of death in the US, after heart disease. A total of 1.9 million new cancer cases and 609,360 deaths from cancer are expected to occur in the US in 2022, which is about 1,670 deaths a day.

There is still no permanent cure to cure cancer, early detection is crucial for treatment and survival of patient.

Image recognition and deep learning have been used effectively in detection and treatment of several dangerous diseases, helping in early diagnosis and treatment.

The risk of death from cancer dropped by about 2% a year from 2015 through 2019 compared to 1% a year during the 1990s. Accelerating declines in the cancer death rate show the power of prevention, screening, early diagnosis and treatment.

Deep learning can be used to analyze features allowing detection of breast cancer.

Two of the most common imaging used in breast cancer detection are histopathology and mammography.

In our research we aim to compare and use different CNN architectures (based on densenet-121 and inception v-4) on each type of the two imaging, while analyzing the results of the different methods for detection and classification.

Key Words: Breast Cancer; Image recognition; Deep Learning; classification; CNN architectures; Inception v-4; densenet-121

**1. Introduction**

Breast cancer is the second most diagnosed cancer worldwide.[1]

Breast cancer occurs in four main types: normal, benign, in-situ carcinoma and invasive carcinoma [2].

In situ carcinoma, the cancer does not affect other organs other than mammary duct lobule system. Benign is not classified as a harmful cancer and involves a minor change in the breast structure. Invasive carcinoma is the deadliest type out of all the four main breast cancer types because it can spread out to all other organs.   
Breast cancer can be diagnosed using one of two approaches: histopathological image analysis or mammography.

Histopathological images are microscopic images of breast tissue that are extremely useful in early treatment of the cancer.  
Mammography is specialized medical imaging that uses a low-dose x-ray system to see inside the breasts. A mammography exam, called a mammogram, aids in the early detection and diagnosis of breast diseases in women.

The main difference between:  
mammography is an earlier type of imaging, before breast tissue is collected for histopathology, an x ray image inside the breasts allows us to search for lumps indicating cancer cells, if there is an indication for breast cancer, breast tissue is collected for analysis under microscope for more accurate diagnosis. [3]

We aim to compare and use 2 models that will satisfy with accuracy for each type while analyzing the results (based on densenet-121 and inception v-4) [4] [5].

we will use each CNN architectures on mammography and histopathology datasets (we will also make use of data augmentation to make our datasets bigger and improv variety).

As well we will study each architecture (Inception V3, DenseNet121) and make a comparison based on the structure aspects of the models and how each of them compares to the two types of images.

**2 Background**

**2.1 Background – ANN**  
These are computing systems inspired by the biological neural networks in animal's minds.  
An ANN is based on a collection of connected units or nodes called artificial neurons (nerve cell**)**, which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit a signal to other neurons. An artificial neuron receives signals then processes them and can signal neurons connected to it. The "signal" at a connection is a real number, and the output of each neuron is computed by some non-linear function of the sum of its inputs. The connections are called edges. Neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Neurons may have a threshold such that a signal is sent only if the aggregate signal crosses that threshold.

Typically, neurons are aggregated into layers. Different layers may perform different transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times.  
  
With this the artificial neural network can perform learning through adjustment of the weights after testing the results of the model.

**2.1.1 Weights**

Each neuron in a neural network computes an output value by applying a specific function to the input values received from the receptive field in the previous layer. The function that is applied to the input values is determined by a vector of weights and a bias (typically real numbers). Learning consists of iteratively adjusting these biases and weights.

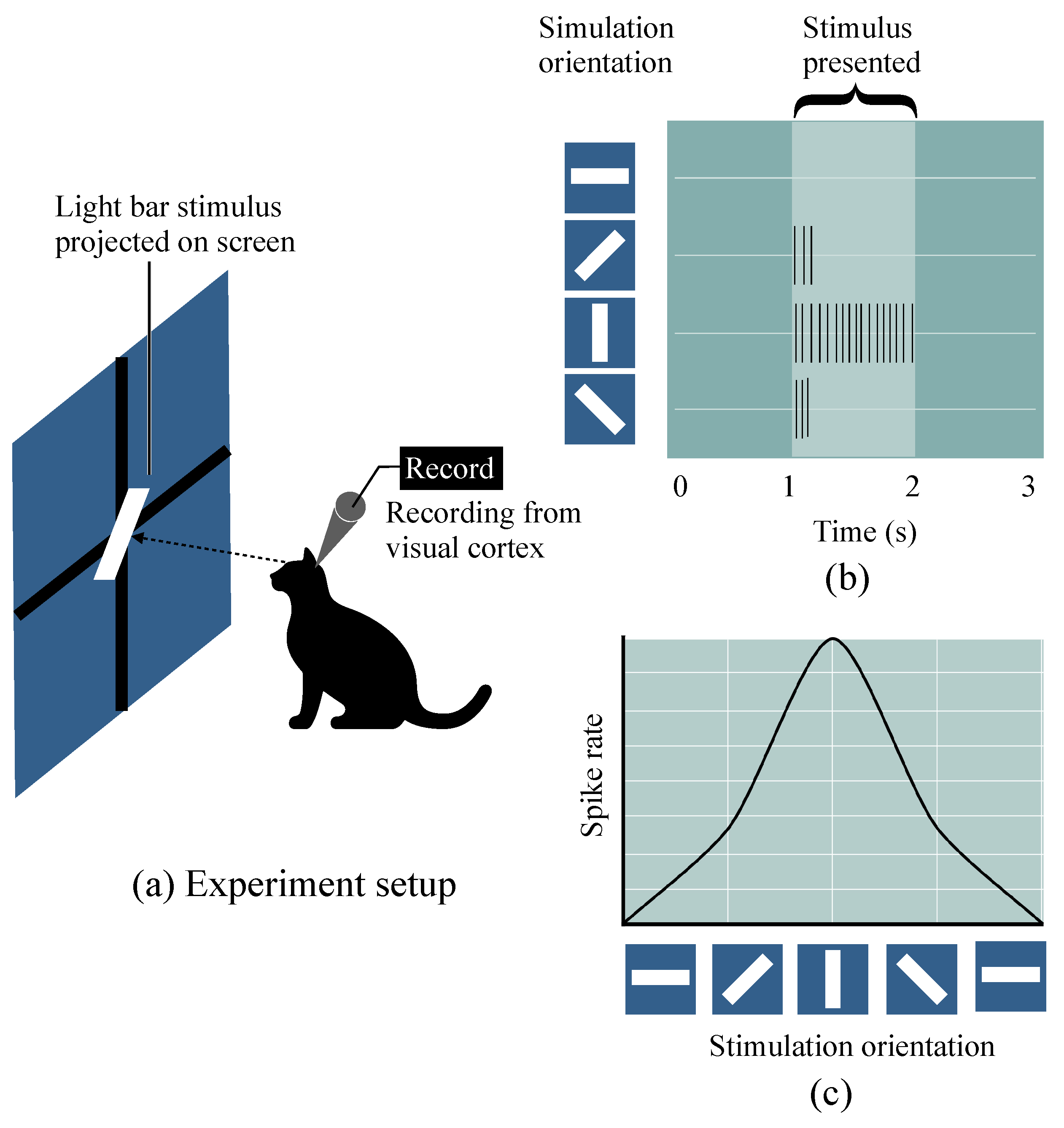
**2.1.2 Activation Function**

In artificial neural networks, the activation function of a node defines the output of that node given an input or set of inputs.  When compared to a neuron-based model in nature, the activation function output is to be inputted to the next neuron.

Diagram, line chart

Description automatically generated

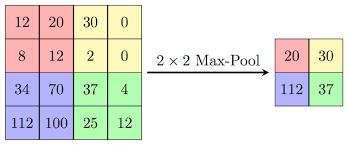
**2.2 CNN (Convolutional neural network)**  
Convolutional neural networks are a specialized type of artificial neural networks that use a mathematical operation called convolution in place of general matrix multiplication in at least one of their layers.   
They are specifically designed to process pixel data and are used in image recognition and processing.[6]

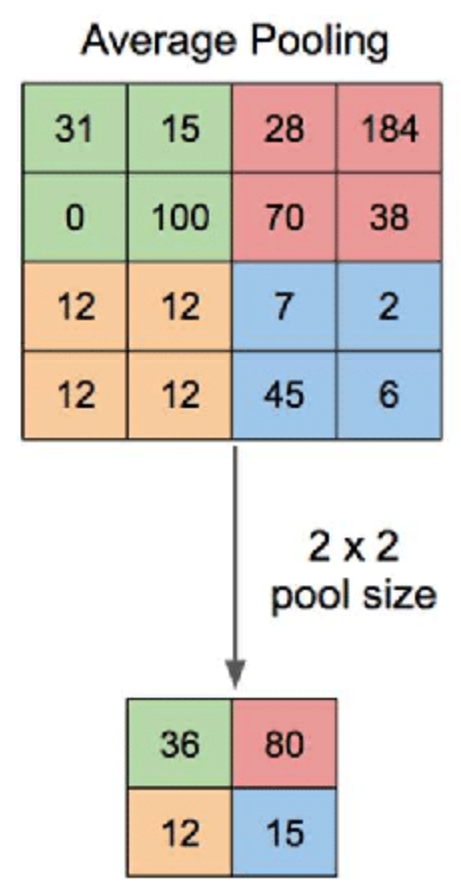
Development of CNN, specifically the type of filters used, was inspired by studies of organization of animal visual cortex such as Hubel & Wiesel.[7]  
  
  
A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution.  
Digital images are stored as pixel values in a two-dimension array.  
In a CNN model we then go over the data with filters (such as filters that search for a certain color or a type of edge) and extract features from the image, and as layers feed onto each other extracted features can become more complex in deeper layers.

**2.2.1 Convolutional layers**   
In a CNN, the input is a tensor with a shape: (number of inputs) × (input height) × (input width) × (input channels). After passing through a convolutional layer, the image becomes abstracted to a feature map, also called an activation map, with shape: (number of inputs) × (feature map height) × (feature map width) × (feature map channels).

Convolutional layers convolve the input and pass its result to the next layer. This is like the response of a neuron in the visual cortex to a specific stimulus.

**2.2.2 pooling layers**  
Pooling layers reduce the dimensions of data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer.  
There are two common types of pooling in popular use: max and average. Max pooling uses the maximum value of each local cluster of neurons in the feature map, while average pooling takes the average value.

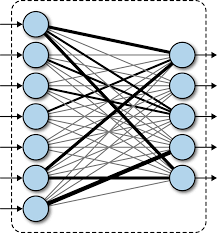




**2.2.3 Fully Connected Layer**

Fully connected layers connect every neuron in one layer to every neuron in another layer.

They come as the last layers after convolution and pooling and these layers purpose is to take the output of previous layers to classify the image into a label (if for instance we have 3 possible layers: car, dog or cat, the last layer will have 3 nodes, and the weights will point each possibility and strength of the 3 labels).



**3. Project Review and Process Description.**

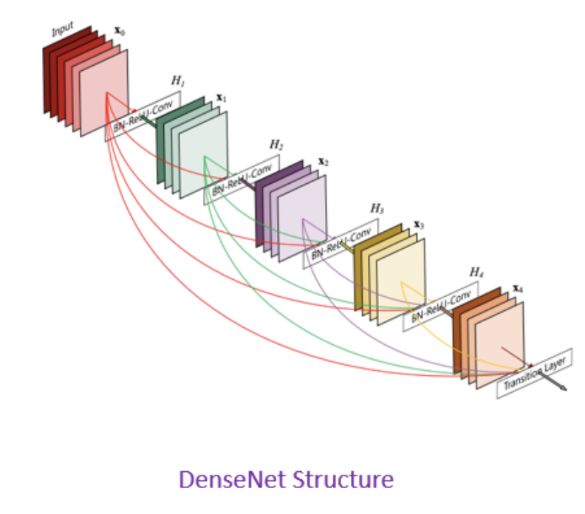
**3.1 Architecture – DenseNet-121 (Densely Connected Convolutional Networks)**

In [4] a traditional feed-forward Convolutional Neural Network (CNN), each convolutional layer except the first one (which takes in the input), receives the output of the previous convolutional layer and produces an output feature map that is then passed on to the next convolutional layer. Therefore, for 'L' layers, there are 'L' direct connections, one between each layer and the next layer.

However, as the number of layers in the CNN increase, i.e., as they get deeper, the '**vanishing gradient**' problem arises. This means that as the path for information from the input to the output layers increases, it can cause certain information to 'vanish' or get lost which reduces the ability of the network to train effectively.

**3.1.1 Vanishing Gradient problem solve**

DenseNets resolve vanishing gradient problem by modifying the standard CNN architecture and simplifying the connectivity pattern between layers. In a DenseNet architecture, each layer is connected directly with every other layer, hence the name Densely Connected Convolutional Network. For 'L' layers, there are L(L+1)/2 direct connections.



**3.1.2 DenseNet Components**

DenseNet components including 4 parts:

* Connectivity
* DenseBlocks
* Growth Rate
* Bottleneck layers

**3.1.2.1 Connectivity**

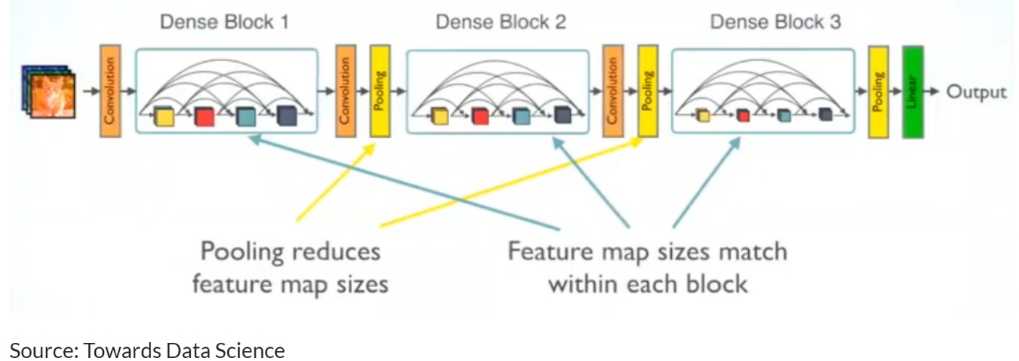
in each layer, the feature maps of all the previous layers are not summed, but concatenated and used as inputs. Consequently, DenseNets require fewer parameters than an equivalent traditional CNN, and this allows for feature reuse as redundant feature maps are discarded. So, the lth layer receives the feature-maps of all preceding layers, x0,...,xl-1, as input:

**3.1.2.2 DenseBlocks**

The use of the concatenation operation is not feasible when the size of feature maps changes. However, an essential part of CNNs is the down-sampling of layers which reduces the size of feature-maps through dimensionality reduction to gain higher computation speeds.

To enable this, DenseNets are divided into DenseBlocks, where the dimensions of the feature maps remains constant within a block, but the number of filters between them is changed. The layers between the blocks are called Transition Layers which reduce the number of channels to half of that of the existing channels.

For each layer, from the equation above, Hl is defined as a composite function which applies three consecutive operations: batch normalization (BN), a rectified linear unit (ReLU) and a convolution (Conv).



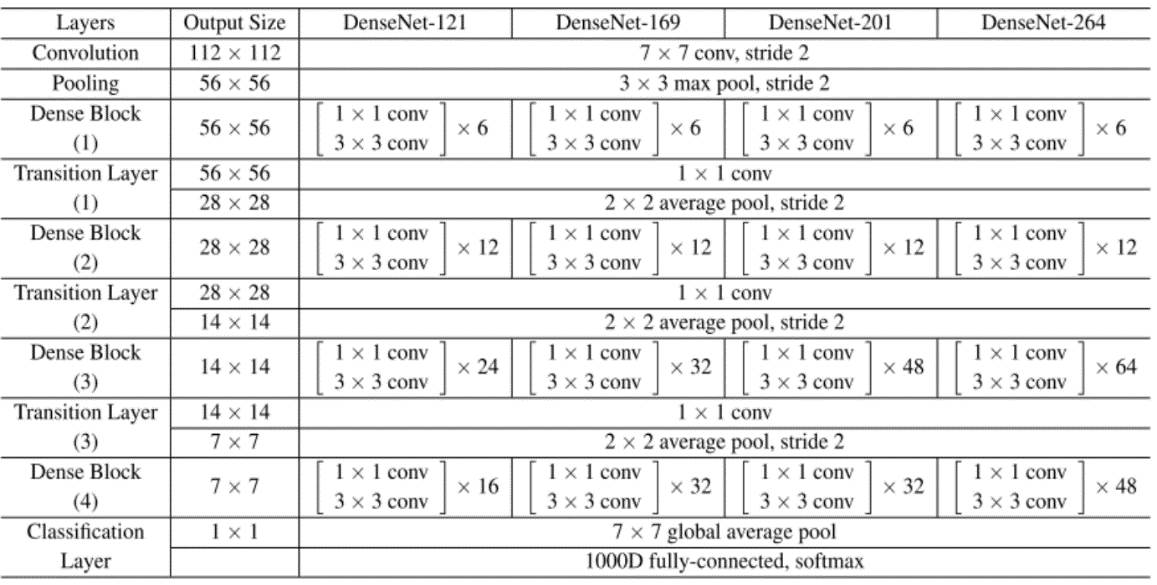
In the above image, a deep DenseNet with three dense blocks is shown. The layers between two adjacent blocks are the transition layers which perform downsampling (i.e. change the size of the feature-maps) via convolution and pooling operations, whilst within the dense block the size of the feature maps is the same to enable feature concatenation.

**3.1.2.3 Growth Rate**

One can think of the features as a global state of the network. The size of the feature map grows after a pass through each dense layer with each layer adding 'K' features on top of the global state (existing features). This parameter 'K' is referred to as the growth rate of the network, which regulates the amount of information added in each layer of the network. If each function H l produces k feature maps, then the lth layer has

**3.1.2.4 Bottleneck layers**

Although each layer only produces k output feature-maps, the number of inputs can be quite high, especially for further layers. Thus, a 1x1 convolution layer can be introduced as a bottleneck layer before each 3x3 convolution to improve the efficiency and speed of computations.

**3.1.3 DenseNet Architecture**

Using the DenseNet-121 architecture to understand the table, we can see that every dense block has varying number of layers (repetitions) featuring two convolutions each; a 1x1 sized kernel as the bottleneck layer and 3x3 kernel to perform the convolution operation.

Also, each transition layer has a 1x1 convolutional layer and a 2x2 average pooling layer with a stride of 2. Thus, the layers present are as follows:

1. Basic convolution layer with 64 filters of size 7X7 and a stride of 2.
2. Basic pooling layer with 3x3 max pooling and a stride of 2.
3. Dense Block 1 with 2 convolutions repeated 6 times.
4. Transition layer 1 (1 Conv + 1 AvgPool).
5. Dense Block 2 with 2 convolutions repeated 12 times.
6. Transition layer 2 (1 Conv + 1 AvgPool).
7. Dense Block 3 with 2 convolutions repeated 24 times.
8. Transition layer 3 (1 Conv + 1 AvgPool).
9. Dense Block 4 with 2 convolutions repeated 16 times.
10. Global Average Pooling layer- accepts all the feature maps of the network to perform classification.
11. Output layer.

Therefore, DenseNet-121 has the following layers:

* 1 7x7 Convolution
* 58 3x3 Convolution
* 61 1x1 Convolution
* 4 AvgPool
* 1 Fully Connected Layer

To summarize DenseNet-121 has 120 Convolutions and 4 AvgPool.

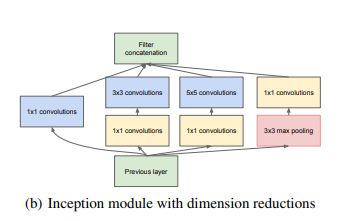
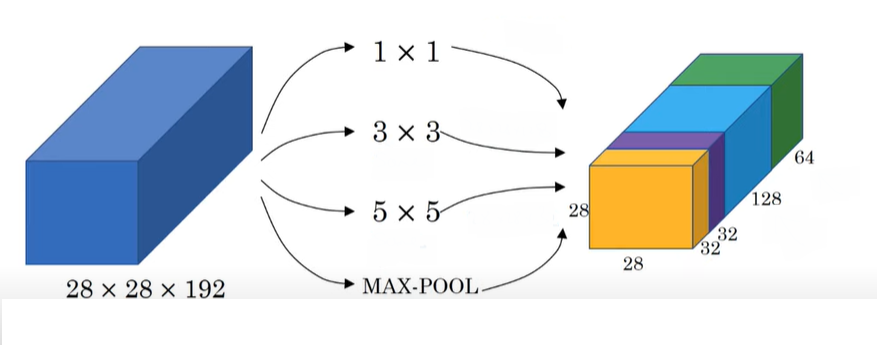
All layers i.e., those within the same dense block and transition layers, spread their weights over multiple inputs which allows deeper layers to use features extracted early on.

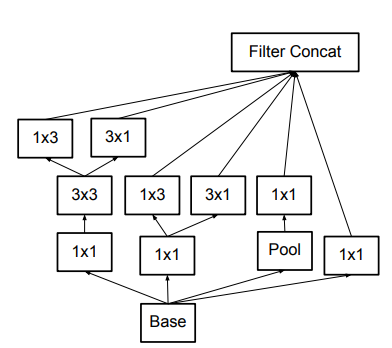
**3.1.4 Advantages of the DenseNet**

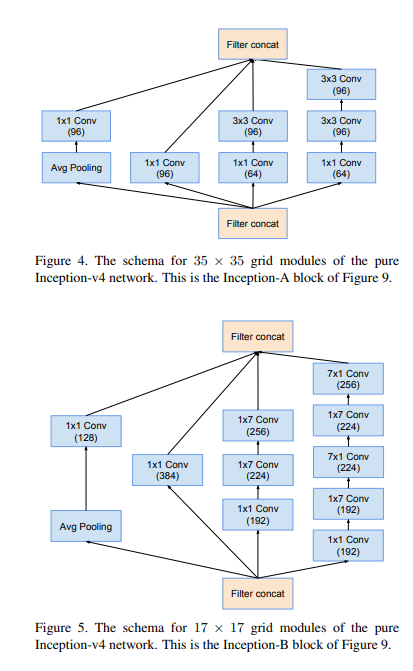
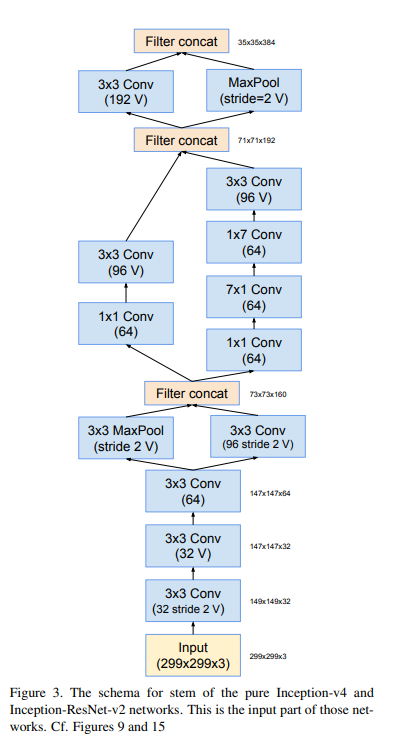
Two of the most obvious Advantages of the DenseNet are

**Parameter efficiency** and **Implicit deep supervision** which result in more compact models and have achieved state of the art of performances and better results across competitive datasets, as compared to their standard CNN counterpart.

* **Parameter efficiency** – Every layer adds only a limited number of parameters- for e.g. only about 12 kernels are learned per layer
* **Implicit deep supervision** – Improved flow of gradient through the network- Feature maps in all layers have direct access to the loss function and its gradient.

**3.2 Architecture - Inception**  
In [5] the inception network which was introduced in [11] Szegedy et al. 2015a takes a new approach to deep layers models where before different types of convolutions and pooling layers were stacked upon each other but now in an inception module we combine within the same layer different convolutions and a pooling operation which all produces at the layer level an output with same dimensions only different depths which are than concatenated as an output of the layer and passed on unto the next.  
  
inception was made with a goal to reduce parameters needed for the model and at its publication was able to produce better results than former naïve version with much less parameters with 1x1 convolutions.

inception v2 introduced new modules variations and the use of auxiliary classifiers [12]  


Inception v3 is also introduced in the same paper with v2 and is a variation of v2 with batch normalization on the auxiliary classifiers and not just on convolutions.  
the Inception architecture is highly tunable, meaning that there are a lot of possible changes to the number of filters in the various layers that do not affect the quality of the fully trained network.  
inception v3 is an updated version of inception which made use of advancements and newer tools from when v2 was published.  


**3.3 Research process**

We researched two different CNN models on two different datasets (mammography and histopathology), with various learning rates, epochs, dropouts, and batch sizes.  
We used Kaggle and Google colab that provided us services to run projects with big amount of data. We brought 2 datasets from Kaggle, one for mammography in tf records format and one for histopathology in PNG format.

**Mammography (DDSM) dataset contains the following:**

* contains 55,890 training examples.
* 14% are positive and the remaining 86% negative.
* divided into 5 tfrecords files.
* Negative and positive imagines resized to 299x299.

**Histopathology (BreakHis) dataset contains the following:**

* contains microscopic biopsy images benign and malignant breast tumors.
* 7909 breast cancer histopathology images acquired on 82 patients.
* partial samples of 400x optical zoom images.

The hyperparameters evaluated are epochs (50,100,150), batch size (64, 128), learning rate (0.0005, 0.000005) with a dropout of 0.2.

The programing language that was used to write the system is Python.

**We used the following libraries for our system:**

* pydicom, used to read the data (.dcm format).
* keras, used for the CNN models, layers, loss, and optimizer.
* os, used loading the data from the folders.
* sklearn, used for the train test data split.
* matplotlib, used to plot the accuracy and loss graphs.
* Pyqt5, used for the graphic user interface

We defined the tested parameters this way:

* True Positive, model prediction was positive and data of breast tissue is positive
* True Negative, model prediction was there is no cancer and data of breast tissue is negative
* False Positive, model prediction was positive, and the label of breast tissue was classified there as negative.
* False Negative, model prediction was there is no cancer, and the label was classified as positive.
* Accuracy = (tp+tn)/(tp+ tn+ fp+ fn)
* Precision tp/(tp+ fp)

The number of positive class predictions that belong to the positive class.?

* Recall tp/(tp+ fn)

The number of correct positive predictions made from all positive predictions that could have been made.

**3.4 Results**

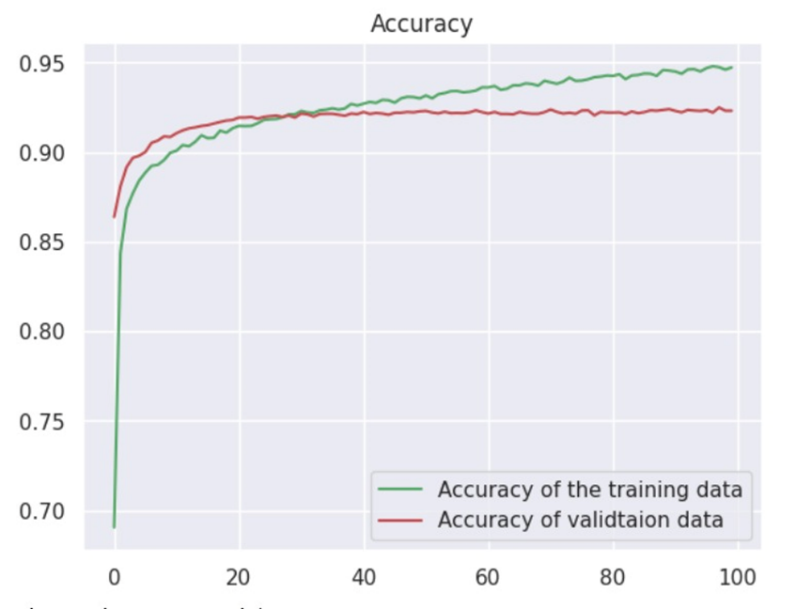
In the tables below there are the training (train\_loss, train\_accuracy), and testing (test\_loss, test\_accuracy) results. With those parameters we can see the results of each model in each phase. The tables indicate which hyperparameters impacted the the models and what is the type of data (filter) that gave us the best over all results. From the tables below after calculated and taking in consideration all of our models we found out that the best overall model for either Mammography and histopathology is the Densenet121 architecture, We concluded that the hyperparameters which had the highest impact on our model and mark them with yellow color which you can see below giving us a better statement about the best parameters for each model with given results.

* T\_acc = Training accuracy
* T\_loss = Training loss
* Test\_acc = Testing accuracy
* Test\_loss = Testing loss

**3.4.1 Mammography**

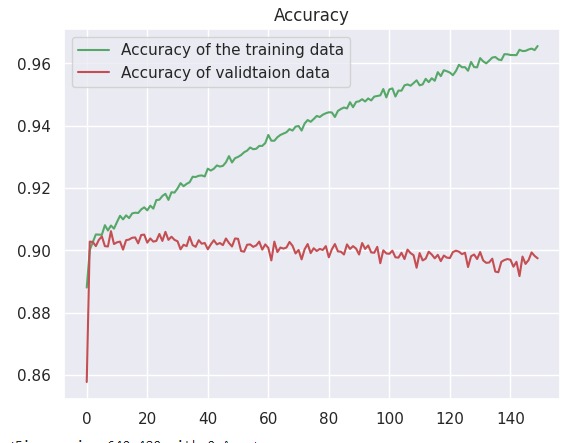
**Densenet121**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test\_acc** | **Test\_loss** | **T\_acc** | **T\_loss** | **model** |
| 0.9269 | 0.3043 | 0.9769 | 0.0858 | B 128 L 0.0005 D 0.2 E 50 |
| 0.9262 | 0.3784 | 0.9814 | 0.0879 | B 128 L 0.0005 D 0.2 E 100 |
| 0.9264 | 0.4066 | 0.9849 | 0.0844 | B 128 L 0.0005 D 0.2 E 150 |
| 0.9255 | 0.1860 | 0.9386 | 0.1503 | B 128 L 0.000005 D 0.2 E 50 |
| 0.9300 | 0.1798 | 0.953 | 0.1182 | B 128 L 0.000005 D 0.2 E 100 |
| 0.9239 | 0.1962 | 0.9647 | 0.0973 | B 128 L 0.000005 D 0.2 E 150 |
| 0.9271 | 0.2902 | 0.97551 | 0.0855 | B 64 L 0.0005 D 0.2 E 50 |
| 0.9304 | 0.3472 | 0.9817 | 0.0907 | B 64 L 0.0005 D 0.2 E 100 |
| 0.9304 | 0.4304 | 0.9839 | 0.1024 | B 64 L 0.0005 D 0.2 E 150 |
| 0.9258 | 0.1822 | 0.9392 | 0.1487 | B 64 L 0.000005 D 0.2 E 50 |
| 0.9306 | 0.1791 | 0.9569 | 0.1078 | B 64 L 0.000005 D 0.2 E 100 |
| 0.9274 | 0.2014 | 0.9669 | 0.0888 | B 64 L 0.000005 D 0.2 E 150 |



**Inception V3**

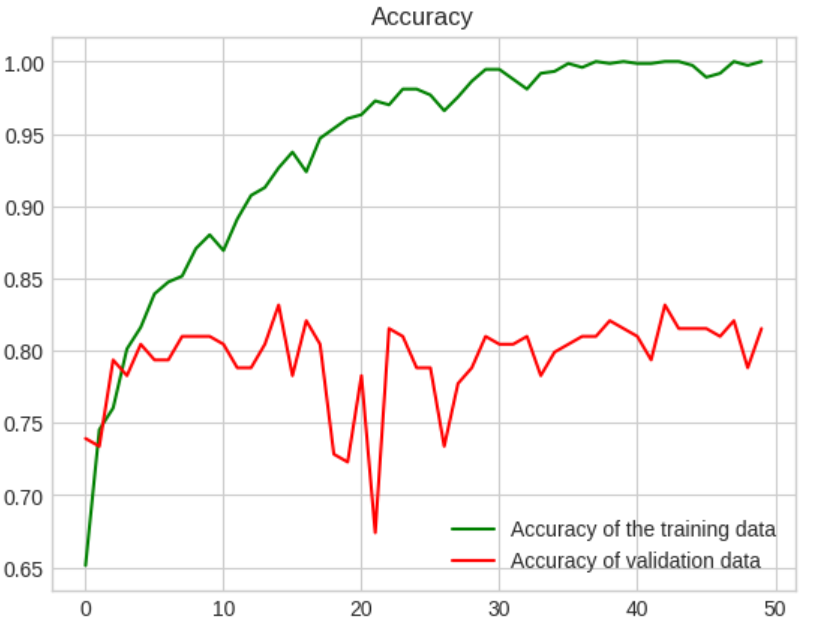
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test\_acc** | **Test\_loss** | **T\_acc** | **T\_loss** | **model** |
| 0.9009 | 0.2701 | 0.9314 | 0.1627 | B 128 L 0.0005 D 0.2 E 50 |
| 0.9026 | 0.3481 | 0.9498 | 0.1348 | B 128 L 0.0005 D 0.2 E 100 |
| 0.9012 | 0.4104 | 0.9656 | 0.1183 | B 128 L 0.0005 D 0.2 E 150 |
| 0.9043 | 0.2573 | 0.9045 | 0.2487 | B 128 L 0.000005 D 0.2 E 50 |
| 0.9018 | 0.2559 | 0.9045 | 0.2403 | B 128 L 0.000005 D 0.2 E 100 |
| 0.9015 | 0.2508 | 0.9053 | 0.2290 | B 128 L 0.000005 D 0.2 E 150 |
| 0.9070 | 0.2614 | 0.9308 | 0.1648 | B 64 L 0.0005 D 0.2 E 50 |
| 0.9044 | 0.3242 | 0.9506 | 0.1325 | B 64 L 0.0005 D 0.2 E 100 |
| 0.8948 | 0.4023 | 0.9645 | 0.1221 | B 64 L 0.0005 D 0.2 E 150 |
| 0.8959 | 0.2593 | 0.9046 | 0.2371 | B 64 L 0.000005 D 0.2 E 50 |
| 0.8977 | 0.2572 | 0.9057 | 0.2281 | B 64 L 0.000005 D 0.2 E 100 |
| 0.8973 | 0.2634 | 0.9063 | 0.2269 | B 64 L 0.000005 D 0.2 E 150 |

****

**3.4.2 Histopathology**

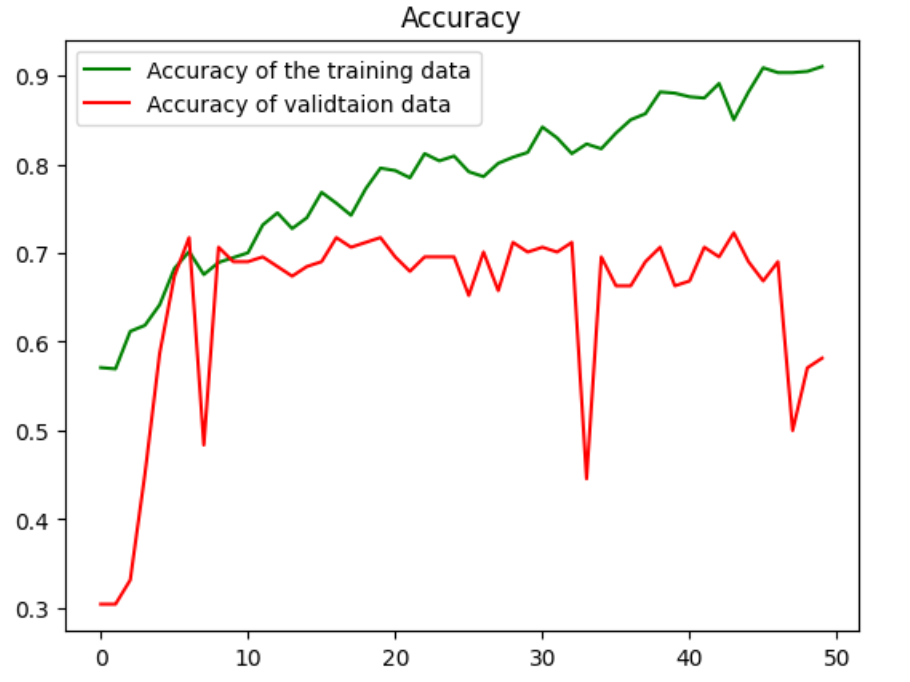
**Densenet121**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test\_acc** | **Test\_loss** | **T\_acc** | **T\_loss** | **model** |
| 0.8440 | 0.4211 | 0.9694 | 0.0963 | B 128 L 0.0005 D 0.2 E 50 |
| 0.8532 | 0.6139 | 0.9760 | 0.0897 | B 128 L 0.0005 D 0.2 E 100 |
| 0.8532 | 0.6767 | 0.9749 | 0.0966 | B 128 L 0.0005 D 0.2 E 150 |
| 0.7724 | 0.5419 | 0.7995 | 0.5143 | B 128 L 0.000005 D 0.2 E 50 |
| 0.7816 | 0.4982 | 0.8104 | 0.4475 | B 128 L 0.000005 D 0.2 E 100 |
| 0.7706 | 0.5099 | 0.8061 | 0.4426 | B 128 L 0.000005 D 0.2 E 150 |
| 0.8623 | 0.5056 | 0.9803 | 0.0745 | B 64 L 0.0005 D 0.2 E 50 |
| 0.8330 | 0.7961 | 0.9705 | 0.1170 | B 64 L 0.0005 D 0.2 E 100 |
| 0.8513 | 0.8260 | 0.9803 | 0.1022 | B 64 L 0.0005 D 0.2 E 150 |
| 0.7834 | 0.5072 | 0.8213 | 0.4694 | B 64 L 0.000005 D 0.2 E 50 |
| 0.7743 | 0.5227 | 0.8453 | 0.4380 | B 64 L 0.000005 D 0.2 E 100 |
| 0.7853 | 0.48296 | 0.8235 | 0.4029 | B 64 L 0.000005 D 0.2 E 150 |

****

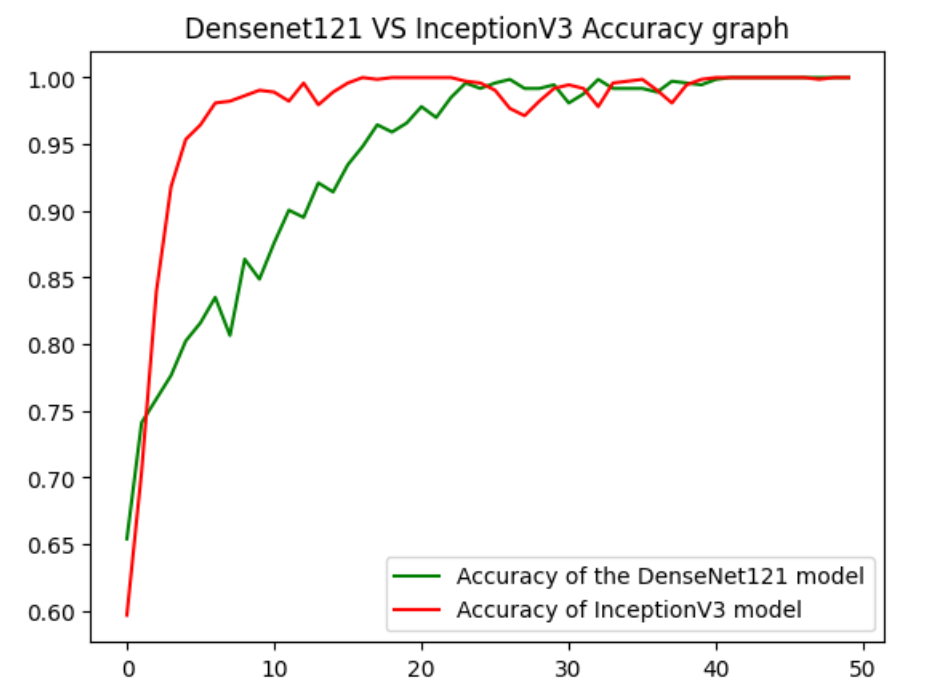
**Inception V3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test\_acc** | **Test\_loss** | **T\_acc** | **T\_loss** | **model** |
| 0.6862 | 0.6274 | 0.8169 | 0.3902 | B 128 L 0.0005 D 0.2 E 50 |
| 0.6678 | 0.8524 | 0.9389 | 0.1900 | B 128 L 0.0005 D 0.2 E 100 |
| 0.6550 | 1.3179 | 0.9389 | 0.2514 | B 128 L 0.0005 D 0.2 E 150 |
| 0.5633 | 0.6907 | 0.6644 | 0.6486 | B 128 L 0.000005 D 0.2 E 50 |
| 0.6403 | 0.6577 | 0.6699 | 0.6699 | B 128 L 0.000005 D 0.2 E 100 |
| 0.6422 | 0.6695 | 0.6742 | 0.6145 | B 128 L 0.000005 D 0.2 E 150 |
| 0.6862 | 0.5701 | 0.8431 | 0.4005 | B 64 L 0.0005 D 0.2 E 50 |
| 0.6825 | 0.7997 | 0.8681 | 0.3081 | B 64 L 0.0005 D 0.2 E 100 |
| 0.6403 | 1.2270 | 0.9324 | 0.2173 | B 64 L 0.0005 D 0.2 E 150 |
| 0.6165 | 0.6945 | 0.6437 | 0.6616 | B 64 L 0.000005 D 0.2 E 50 |
| 0.6477 | 0.7011 | 0.6775 | 0.6415 | B 64 L 0.000005 D 0.2 E 100 |
| 0.6605 | 0.6610 | 0.7069 | 0.5898 | B 64 L 0.000005 D 0.2 E 150 |

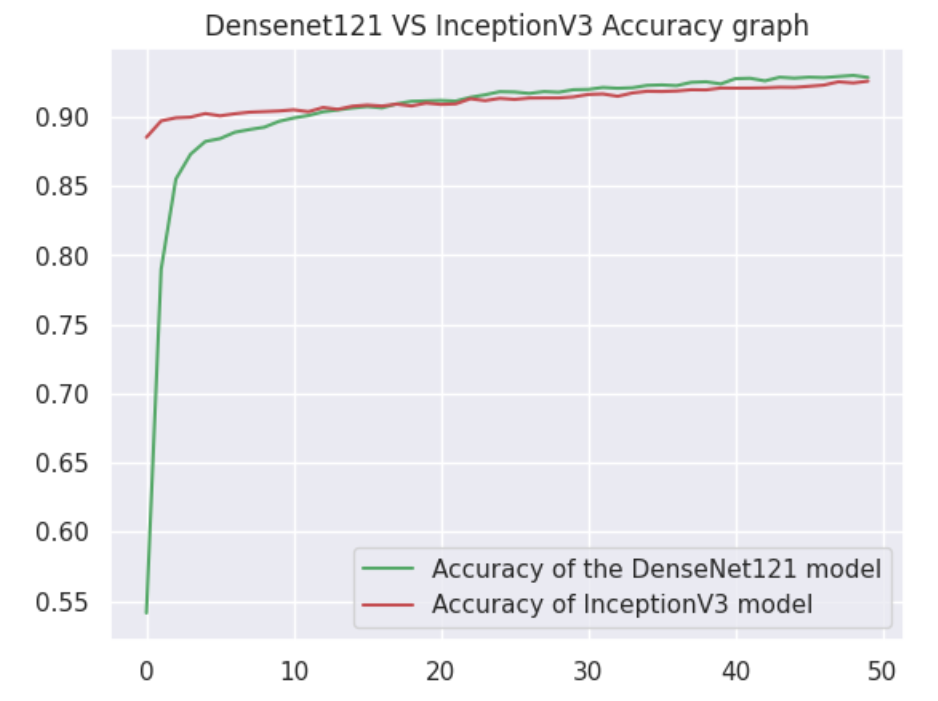


**3.4.3 Densenet 121 VS Inception V3**

**Histopathology**



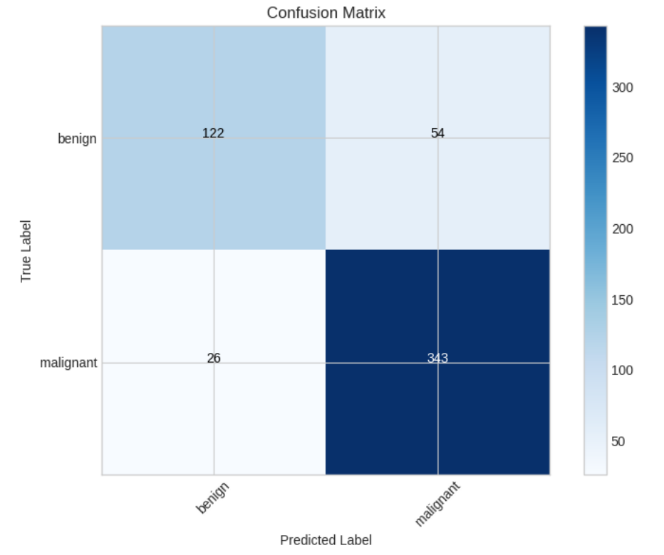
**Mammography**



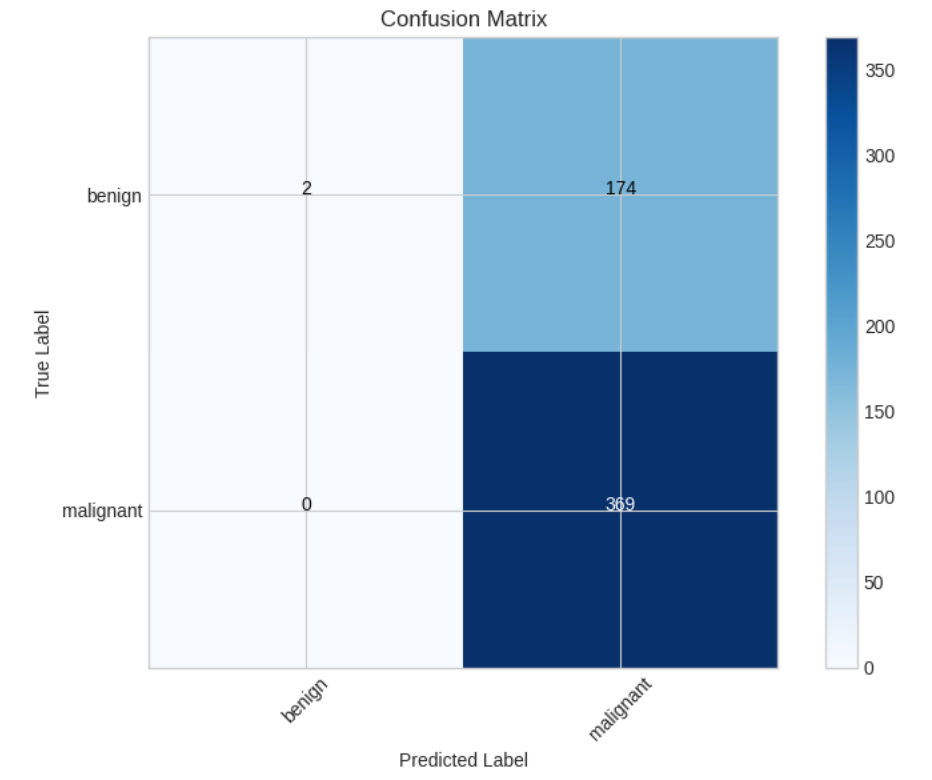
**3.4.4 Confusion Matrix**

We take into consideration couple of measures that the confusion matrix provide us, such as accuracy, precision, recall and F1-score.

DenseNet121



Inception – V3

****

Measures values that we consider in our comparison between the models:

DenseNet\_Accuracy: 0.8532 Inception\_Accuracy: 0.6807

DenseNet\_Precision: 0.8640 Inception\_Precision: 0.6796

DenseNet\_Recall: 0.9295 Inception\_Recall: 1.0000

DenseNet\_F1 Score: 0.8956 Inception\_F1 Score: 0.8092

**3.4.5 Discussions and Conclusions**

We have compared 2 models on 2 different datasets to measure which model will give the better results while considering different aspects of the datasets such as Mammography which is easier to detect if there is a breast cancer in comparison to Histopathological images.

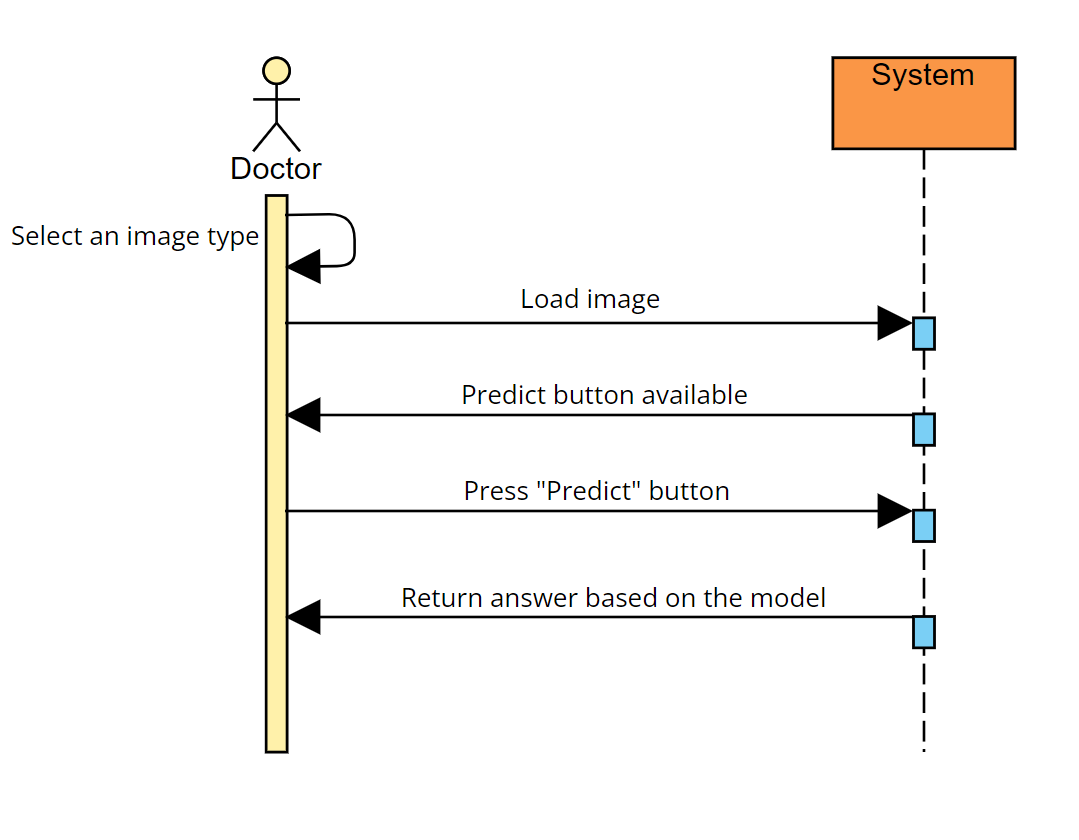
After testing the models on different datasets with different hyperparameters we reach a conclusion that DenseNet 121 model perform better then InceptionV3 on both of the datasets.

It is important to say that by our graphs we can see that the Inception model did extraordinary well for the first couple of epochs but after a certain point DenseNet reach a higher point with the longer we run our models (more epochs).

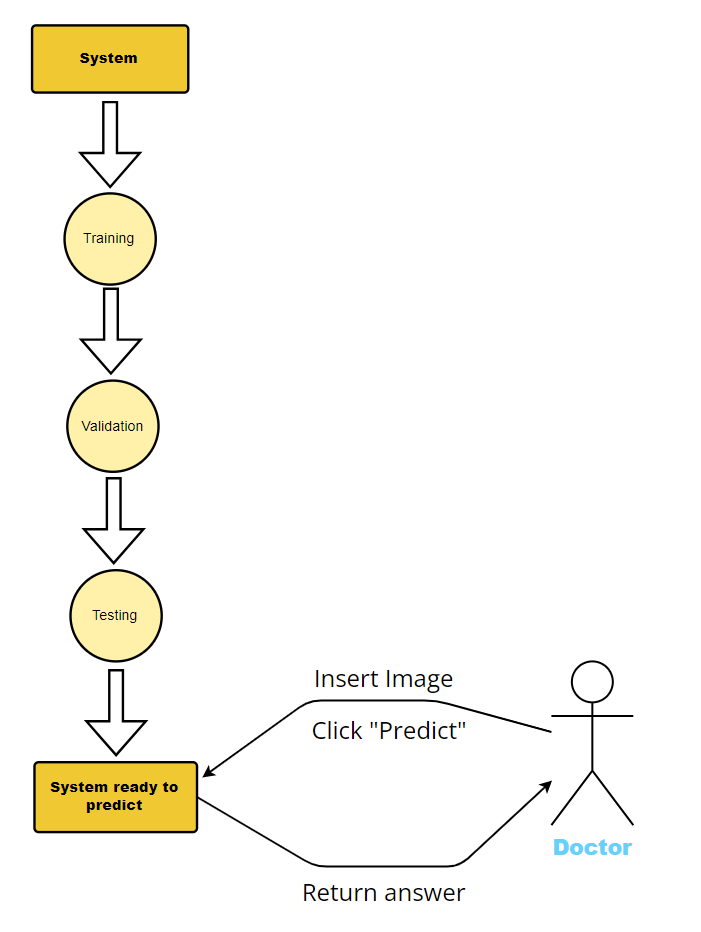
We hope our project showcase that it is possible to use image recognition and patterns identifying using state of art models to use these tools to help medical professionals with decisions making regrades medical conditions.

**4 Product**

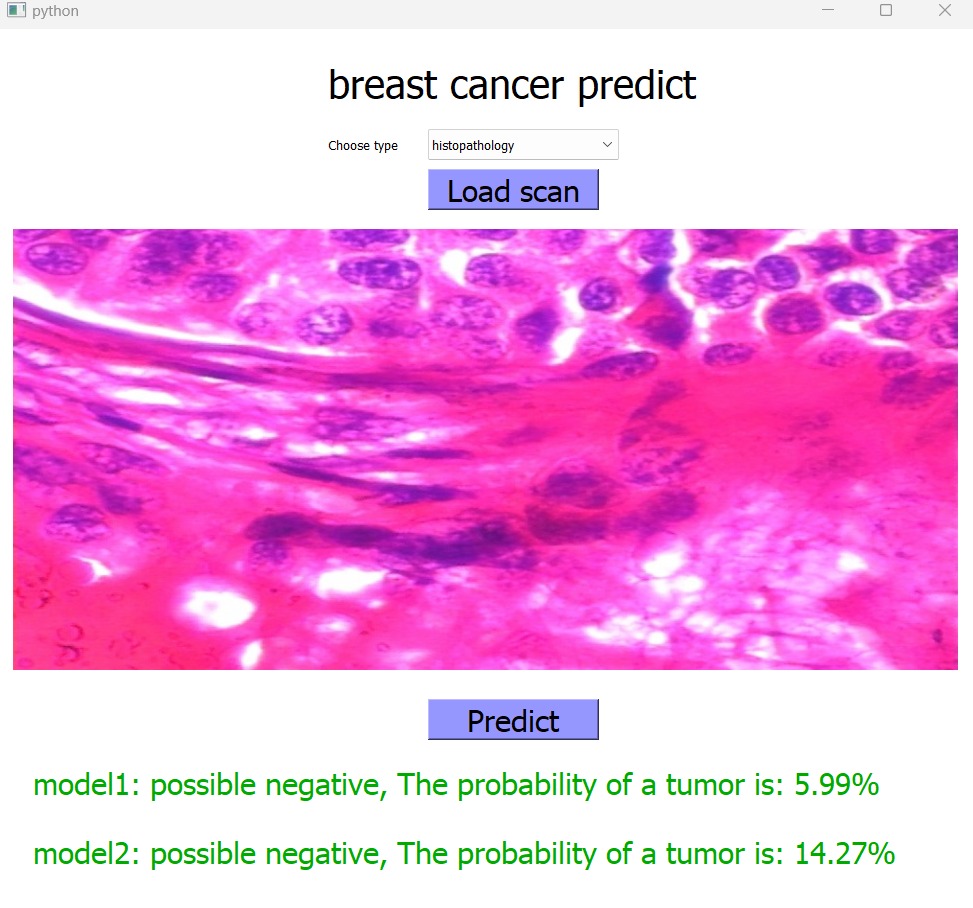
**4.1 Sequence diagram**



**4.2 Flowchart diagram**



**4.3 GUI**

****

**4.4 Description of the operating environment of the system**

**Google Collab:**

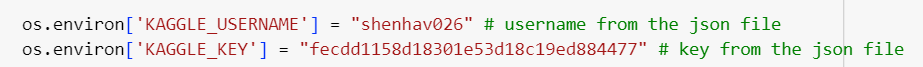
**1.** Enter one of the google collab links (Mammography, Histopathological).

**2.** Sign in with your Gmail account.

**3.** Sign up / Sign in with your Kaggle account and get access to the dataset.

**4.** Change the directory where you want to save the model.

**5.** Run the collab notebook with GPU option (if accessible).

Generate the key from the json file(support by Kaggle)

Change to the directory in this line(DenseNet example, need to be changed in Inception as well)

**Application:**

**1.** Load the python file in the ide with the entire directory (contains the model)

**2.** Follow the README instruction in the Github to download the models and save the models in their correct path in the code.



**3.** Run the code.

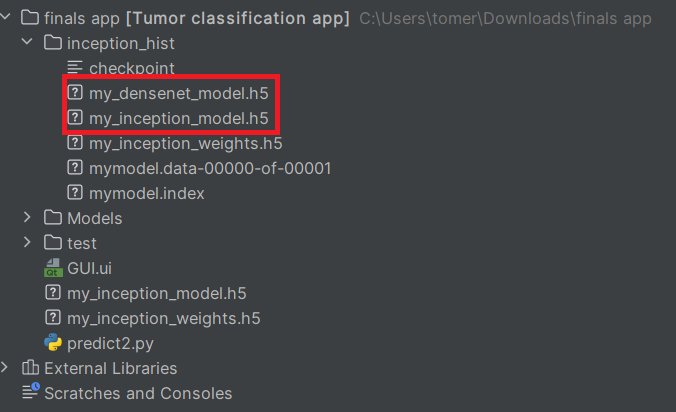
**4.** Choose the Load scan button and select one of the medical scans in PNG format.

**5.** Wait for the picture to load to the screen.

**6.** Press predict button.

**7.** Model 1 result classify as DenseNet121, Model 2 result classify as InceptionV3.

You can see the entire directory as it should look like including both of the models.

****

Google Collab: [Mammography Model](https://colab.research.google.com/drive/12VHK-C_J8ryHk79MxJEpe8DbFzQvoWJO?usp=sharing) , [Histopathology Model](https://colab.research.google.com/drive/1LH7VFgw6zSBn2Bxm8EzI2CUi98vNN9i8?usp=sharing) .

**5 Reference**

[1] S. Germano and L. O'Driscoll, Curr. Cancer Drug Targets, 2009, 9, 398–418.

[2] “Endometrial Cancer - Symptoms and Causes.” Mayo Clinic, 20 May 2021, [www.mayoclinic.org/diseases-conditions/endometrial-cancer/symptoms-causes/syc-20352461](http://www.mayoclinic.org/diseases-conditions/endometrial-cancer/symptoms-causes/syc-20352461).

[3] Radiological Society of North America (RSNA) and American College of Radiology (ACR). “Mammography.” Radiologyinfo.org, [www.radiologyinfo.org/en/info/mammo](http://www.radiologyinfo.org/en/info/mammo).

[4] Huang, Gao. “Densely Connected Convolutional Networks.” arXiv.org, 25 Aug. 2016, arxiv.org/abs/1608.06993.

[5] Szegedy, Christian. “Inception-v4, Inception-ResNet and the Impact of Residual...” arXiv.org, 23 Feb. 2016, arxiv.org/abs/1602.07261v2.

[6] Ian Goodfellow and Yoshua Bengio and Aaron Courville (2016). [Deep Learning](https://www.deeplearningbook.org/). MIT Press. p. 326.

[7] “Brain Mechanisms of Vision.” Scientific American, [www.scientificamerican.com/article/brain-mechanisms-of-vision/?error=cookies\_not\_supported&code=6e167a00-2a43-4b05-977e-5564c21f63a9](http://www.scientificamerican.com/article/brain-mechanisms-of-vision/?error=cookies_not_supported&code=6e167a00-2a43-4b05-977e-5564c21f63a9).

[8] Alom MZ, Yakopcic C, Nasrin MS, Taha TM, Asari VK. Breast Cancer Classification from Histopathological Images with Inception Recurrent Residual Convolutional Neural Network. J Digit Imaging. 2019 Aug;32(4):605-617. doi: 10.1007/s10278-019-00182-7. PMID: 30756265; PMCID: PMC6646497.

[9] Ziliang Zhong et al 2020 J. Phys.: Conf. Ser. 1651 012143

[10] Altameem, Ayman. “Breast Cancer Detection in Mammography Images Using Deep Convolutional Neural Networks and Fuzzy Ensemble Modeling Techniques.” MDPI, [www.mdpi.com/2075-4418/12/8/1812](http://www.mdpi.com/2075-4418/12/8/1812).

[11] Going Deeper With Convolutions.” arXiv.org, 17 Sept. 2014, arxiv.org/abs/1409.4842.

[12] “Rethinking the Inception Architecture for Computer Vision.” arXiv.org, 2 Dec. 2015, arxiv.org/abs/1512.00567.

[13] RSNA Screening Mammography Breast Cancer Detection | Kaggle. [www.kaggle.com/competitions/rsna-breast-cancer-detection/data](http://www.kaggle.com/competitions/rsna-breast-cancer-detection/data).

[14] Spanhol, F., Oliveira, L. S., Petitjean, C., Heutte, L., A Dataset for Breast Cancer Histopathological Image Classification, IEEE Transactions on Biomedical Engineering (TBME), 63(7):1455-1462, 2016.