BREAST IMAGE CLASSIFICATION USING ENSEMBLE DEEP LEARNING ARCHITECTURES

A THESIS SUBMITTED TO

THE FACULTY OF ARCHITECTURE AND ENGINEERING

OF

EPOKA UNIVERSITY

BY

ARTJOLA GANELLARI

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR

THE DEGREE OF MASTER OF SCIENCE

IN

COMPUTER ENGINEERING

JUNE, 2020

Approval sheet of the Thesis

This is to certify that we have read this thesis entitled **"Breast Image Classification using Ensemble Deep Learning architectures"** and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Dr. Ali Osman Topal
Head of Department
Date: June 26th ,2020

Examining Committee Members:

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Artjola Ganellari

Signature: _____

ABSTRACT

BREAST IMAGE CLASSIFICATION USING ENSEMBLE DEEP LEARNING ARCHITECTURES

Ganellari, Artjola

M.Sc., Department of Computer Engineering

Supervisor: Dr. Maaruf Ali

Breast cancer is one of the most widespread diseases around the world which mostly leads to the death not only in women, but also men. The rapid development of the technology and the disruption era, have contributed into finding feasible solutions to diagnose this type of cancer in its early stages. These have made possible the acquirement of high-resolution breast tissue images. On the other hand, Computed Aided Detection (CAD) Systems have taken advantage of the availability of the datasets, and have somehow overcame the struggles in diagnosing the breast cancer. Being difficult to analyze because of their shape, size, and different contrast levels, breast tissues classifications cannot be utilized by the manual or traditional image processing techniques. Considering the afore-mentioned reasons, there is room for computerized implementations for the classification of the breast tissues into malignant or benign.

Convolutional Neural Networks (CNN) have assisted in the classification of the abnormalities, producing very accurate results. In this study, we examine the ability of such networks in classifying the breast medical images into two categories: benign or malign. This approach, adapts two pre-trained deep learning architectures such as Densenet201 and VGG16 and ensemble them in a stack learning model. Each of the architectures has been trained separately, and then the outputs of them both, have been fed into a stack model of Multilayer perceptron classifier in order to proceed with the classification of the tissues. For this task, we have used a publicly available dataset of breast histology called BreakHis. It contains a total of 7909 medical images from which 5429 malign and 2480 benign. The proposed ensemble model obtained a

higher prediction accuracy than the single implemented classifiers. It achieved accuracy of 91.03%, demonstrating a successful utilization of deep learning architectures in classifying breast tissues.

Key words: *Deep learning, Ensemble learning, classification, breast tissue analysis, Convolutional Neural Networks*

ABSTRAKT

KLASIFIKIMI I IMAZHEVE DIXHITALE TE GJIRIT DUKE PERDORUR ARKITEKTURA TE ENSEMBLUARA TE DEEP LEARNING

Ganellari, Artjola

M.Sc., Department of Computer Engineering

Supervisor: Dr. Maaruf Ali

Kanceri i gjirit është një nga sëmundjet më të përhapura në të gjithë botën e cila në shumicën e rasteve con në rritjen e vdekshmërisë jo vetëm tek femrat por edhe tek meshkujt. Zhvillimi i shpejtë i teknologjisë ka kontribuar ne gjetjen e zgjidhjeve me efektive ne diagnostikimin e ketij loj kanceri ne fazat e para te zhvillimit..Këto kanë bërë të mundur përvetësimin e imazheve të indeve të gjirit me rezolucion të lartë. Nga ana tjetër Computed Aided Detection (CAD) kanë përfituar nga disponueshmëria e të dhënave, dhe kanë kapërcyer disi problemet e hershme gjatë diagnostikimit të kancerit të gjirit. Duke qenë se eshte e vështirë për t'u analizuar për shkak të formës, madhësisë dhe niveleve të ndryshme të kontrastit, klasifikimet e indeve të gjirit nuk mund të përdoren në mënyrë manuale ose duke perdorur teknikat tradicionale të përpunimit të imazhit. Duke marrë parasysh arsyet e lartpërmendura, ekziston nje hapësirë për zbatime të kompjuterizuara.

CNN kane kontribuar ne klasifikimin të indeve të gjirit në malinje ose beninje. CNN kanë ndihmuar në klasifikimin e anomalive, duke dhënë rezultate shumë të sakta. Në këtë studim, ne shqyrtojmë aftësinë e rrjete të tilla në klasifikimin e saktë të imazheve mjekësore të gjirit në dy kategori: beninje ose malinje. Kjo qasje, përshtat dy arkitektura të para-trajnuara për Deep Learning si DenseNet201 dhe VGG16 dhe i bashkon ato në një model te vetem ensemblimi. Secila prej arkitekturave është trajnuar veçmas, dhe më pas rezultatet e të dyjave, janë futur në një kete model ensemblimi duke perdorur një klasifikues si Myltilayer Perceptron në mënyrë që të vazhdohet me klasifikimin e indeve.Për këtë studim , ne kemi përdorur një bazë të dhënash në publikuara me imazhe histologjike së gjirit të quajtur BreakHis. Ajo përmban një total

prej 7909 imazhe mjekësore nga të cilat 5429 janë malinje dhe 2480 inde beninje. Modeli i propozuar ka dhene një saktësi më të lartë parashikimi sesa DenseNet and VGG te trajnuar ne meyre te pavarur. Ky sistem arriti një saktësi prej 91.03% duke demonstruar një përdorim të suksesshëm të arkitekturave të Deep Learning në klasifikimin e indeve të gjirit.

Fjale kyce: *Deep learning, Ensemble learning, klasifikim, analiza te indeve te gjirit, Convolutional Neural Networks*

ACKNOWLEDGEMENTS

I would like to express my special thanks to my supervisor Dr. Maaruf Ali for his continuous guidance, encouragement, motivation and support during all the stages of my thesis. I sincerely appreciate the time and effort he has spent to improve my experience during the last year of my graduate studies.

Table of Contents

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

Breast cancer has become one of the main reasons that cause death to women in a global perspective. If we relate to the statistics of World Health Organizations, in 2018, 627’000 women have died from breast cancer (WHO, n.d.). These rates have substantially increased each year. Considering this, it is obvious that solutions to detect and diagnose early this kind of disease, can be flourished out, offering treatment and care in an effective way.

There are methods, traditional and manual ones, that can distinguish the malign and benign cells on the breast tissues, but it is usually a very complicated task. Several reasons lie behind this, starting from the diverse shapes of the tissues, and continuing with the contrasting levels of the image or the harsh texture of the cell. These kind of challenges, does affect the time factor as well as encouraging the subjectivness of the interpretation by computational brain capacity of humans.

In response to these challenges, with the passing of time, some automatic programs such as CAD systems have been developed in order to overcome the task of the classification of breast medical images. Recently, Deep Learning methods have been established, outperforming most of the literature machine learning techniques for classifying, detecting or segmenting a specific cell structure. Deep learning, in comparison to the other types of automated or manual learning techniques, has the capability to imitate the human learning ability or even surpass it. By extracting the most significant features from each of the image, a CNN model can accurately classify and then learn to predict a part of the dataset that has never be seen from the program.

Ensemble learning on the other hand, is a more complex model, combining some base learning models. Stacking ensemble that will be used in this study, uses as an first level input the predictions of some base leaning models and then uses another model as the second level to predict the final output.

There are several applications that have employed the ensemble learning (Mostafa Amin-Naji, 2019) (Chu, 2019) (Oscar Perdomo, 2019) in the state of the art, and they have achieved high accuracy results on medical images. Inspired by the accomplishments of prior studies, we propose a stack ensemble learning model for binary classification of breast tissue images of BreakHist dataset. The results of the experiments on this study, suggest that the modified ensemble architectures of CNN (DenseNet201 and VGG16) performs better in comparison to the ensemble model of the unmodified architectures.

1.2 Contributions of this paper

In this study, an ensemble deep learning model for the classification of breast tissues, has been demonstrated. More specifically, two different fine tuning pre-trained Convolutional Neural Network models have been modified in order to achieve a higher accuracy in comparison to the other state of the art studies on the same field. Furthermore, the comparison of the accuracy prediction of healthy and non healthy tissues have been employed for both modified and unmodified models. Overall, these model performed well on classifying breast tissues as benign and malign.

1.3 Outline of the thesis

This thesis demonstrate a successful application of an ensemble deep learning model utilized for the binary classification of breast tissue medical images. In this section, we are going to briefly make a summary of each of the upcoming chapters by stating their purpose

What does this thesis demonstrate ? What will each of the chapters contain, describing it with one sentence.

Chapter 2 gives a comprehensive understanding of the chain AI- ML – DL. It describes in detail each of the each of the mentioned technologies while focusing more on how a Convolutional Neural Network works.

Chapter 3 gives and overview of the literature review, what other studies have achieved which are relevant to our field of interest.

Chapter 4 provides a detailed description of the dataset, the preprocessing techniques used and also the explanation of the models architectures. A report of the experiments conducted in this study is part of this chapter.

Chapter 5 as the final chapter of this thesis, discusses the results of the implemented method. It states the limitations of the study as well as the gaps that it has for future improvements and researches.

CHAPTER 2

BACKGROUND

2.1 Background

Medicine is one of the fields which is closely related to the analysis of images in order to diagnose the disease one person has. The recent developments in biology, medicine but also in technology, has led in a combination of these two fields. There have been found gaps between medicine and technology and there have been found solutions that if these fields incorporate together, major changes can be observed in the efficiency of the results of the medical analysis.

With the development of medical images and the creation of large and high-resolution databases of images, the opportunities to make improvements and solve the biological challenges have also increased rapidly. However, it is extremely difficult to process and analyze the medical images because the tissues are of different sizes, shapes, and also cannot be distinguished clearly from the background of the image. Considering these, the overall process of medical image analysis, the traditional methods or the technological ones, make it not just time consuming but also challenging to attain high accuracy. Subsequently, these limitations crave for the development of the new methods to classify and segment the tissues of medical images.

There are a lot of image processing techniques such as machine learning and deep learning. These techniques have remarkably helped in the image processing field by making segmentation of regions of interest, classification of the tissues by contributing in this way in the diagnosis of the disease. To better understand the techniques that have recently been used to process the medical images, below will be presented briefly, their hierarchical representation.

2.2 Artificial Intelligence

Artificial intelligence in computer science in the intelligence shown by the machines. Computer architectures that have the ability to execute jobs of human-level intelligence are told to be artificially intelligent. Visual perception, speech recognition, decision-making, language translation are some of the fields where AI finds application. AI is a deep study of the human brain, how it thinks, learns, makes decisions and how it works, when trying to solve problems. And finally, this study outputs intelligent software systems.

The concept of AI concepts is as old as Greek Mythology. Hephaestus and Pygmalion incorporated the idea of intelligent robots as Talos and artificial beings as Galatea and Pandora. It can be said for sure that this concept is as old as the human imagination, but what has been done concretely for this field?

Artificial Intelligence is thought to have very early starts when researches started making a great contribution to the AI viewpoints and techniques. One of the first attempts to try to resemble the true purpose that AI has today, was the study made by Rudolf Carnap in 1928. What this philosopher aimed at was a computational procedure to extract information from past experiences. This was one of the theories that remained without any implementation but this study has helped future generations, engineers, and researches to design and implement machines and Software that fulfills the purpose of AI. Now, AI is shaping the present and the future of technology with the implementations of Machine Learning and Deep Learning.

2.3 Machine Learning

Machine learning is the utilization of artificial intelligence that awards systems the capacity to detect and improve from training without getting external programming. Machine learning converges on the growth of network applications that can reach data and apply it to determine for their own.

Supervised machine learning algorithms can utilize what has been detected in history to the latest data using labeled samples to forecast coming results. Starting from the study of a distinguished training dataset, the training algorithm provides an inferred function to obtain forecasts about the output contents. The practice is capable

of presenting targets for each unique data after adequate training. The learning algorithm can further analyze its output with the accurate, expected output and detect flaws in order to adjust the model subsequently.

On the other hand, unsupervised machine learning algorithms are applied while the information utilized to train is not classified nor labeled. Unsupervised learning subjects how systems can indicate a function to define an unknown arrangement from unlabeled data. The method doesn't estimate the correct output, but it examines the data and can draw presumptions from datasets to report hidden arrangements from unlabeled data.

In this work, we have used supervised learning by using the Convolutional neural Networks that will be explained later on the subsections of this chapter. This type of learning requires for some types of parameters, that once they are found correctly, they can lead to having the best possible generated model with a high accuracy in prediction.

2.4 Deep learning

This is the part that this paper will focus more as a deep learning algorithm will be used to classify the images of the medical tissue.

Deep learning is a subset of machine learning in artificial intelligence (AI) that has interfaces able to master unsupervised from data that is unstructured or unlabeled. Additionally recognized as deep neural learning or deep neural network. Deep learning, a subset of machine learning, employs a hierarchical level of artificial neural networks to continue the process of machine learning. The artificial neural networks are organized like the human brain, with neuron links attached mutually similar to a network. While universal programs model study with data in a linear way, the hierarchical capacity of deep learning methods allows machines to prepare data with a nonlinear strategy.

Deep learning, is part of the whole AI technology which contains most of the state of the art in the medical image analysis. CNN are encouraged by the human brain and are considered to learn the same way as humans. There is a tremendous similarity

between neurons of deep learning and the neurons of the biology. One thing that makes neural networks differ from the ones of the biology, is the usage of the mathematical functions employed in computers. When multiple neurons are combined together and group into layers, the neural network is formed. In this case, the output of one layer, serves as input for the other layer f the neurons. On the other hand, when a network have several layers, the input, hidden and output layer, the Deep Learning models are presented. A lot of hidden layers, communicating with each other, similarly as the human brain, creates a fully connected multilayer network.

To sum up, the differences on the ANN networks and Deep Leaning one can be demonstrated with the following schemes:

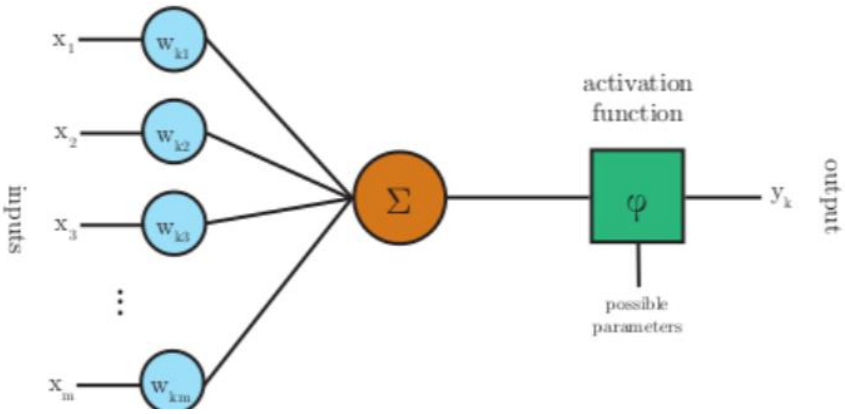

Figure 1 ANN Network

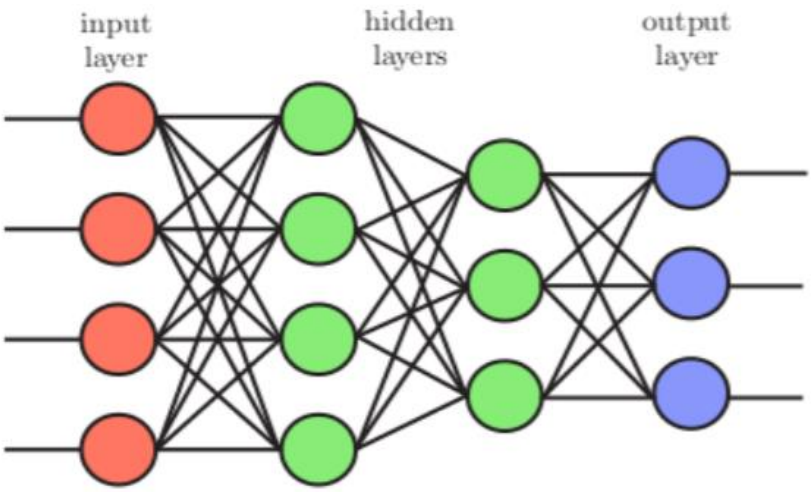

Figure 2 Deep Learning network

2.4.1 Hyperparameters of Deep Learning

In the above sections, we mentioned some parameters that help in constructing a highly performance deep learning model using supervised learning. In this section we are going to explain in detail, what these parameters are.

2.4.1.1 Activation Functions

Activation functions plays a very important role on the construction of deep Learning models. They make decisions of the significance of a neuron, if it should be activated or be left the other way. Once a neuron multiples with its weight and the sum of the total products is found, an activation function is released in order to make a nonlinear transformation to the final value to learn more complex tasks.

Referring to the state of the art works, choosing the best fitting activation function directly affects the model accuracy (P. Ramachandran, 2017) (Matas, 2016).

2.4.1.2 Sigmoid Functions

These types of functions are nonlinear which are mostly applied in feedforward networks. The formula behind this function lies on the rapport:

$$f(x) = \left(\frac{1}{1 + exp^{-x}}\right)$$

As this rapport promise, it exists between 0 and 1. For this reason, this kind of function refers as highly significant when it comes to prediction problems, such as binary classification, logistic regression etc. (C. E. Nwankpa, 2018).

2.4.1.3 ReLu Functions

ReLu function is one of the most widespread functions for deep learning as it has arrived to have great success since its launching in 2010. It is a representative of a liner function an follows a threshold in which each input with values less than 0 becomes equal to 0 and it remains the same for values greater or equal to 0. Being very fast on the speed of the computational power, the ReLu functions have a wide usage on Convolutional Neural Networks.

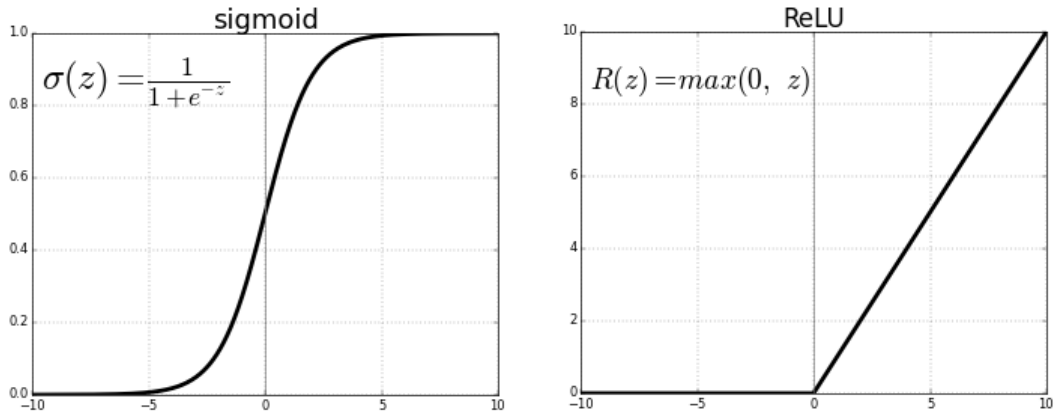

Figure 3 *Sigmoid and ReLu functions*

2.4.1.4 Softmax Functions

Being one other type of activation function, softmax is used to find the probability distribution of from a vector of real numbers. Its output is a range of values from 0 to 1. Its formula is as follows:

$$f(x) = \left(\frac{\exp(xi)}{\sum exp(x_j)}\right)$$

If we had to compare Sigmoid to Softmax functions, the main difference is that the first one is used to binary classification, while the other one for multi class classifications.

2.4.1.4.1 Error functions

Training and testing phases are crucial when developing a specific model, but if it is nor evaluated by using the proper evaluating metrics, it is not worth working on. The loss and Cost function are two critical points when training and testing a model since they determine the quality of the model.

2.4.1.4.2 Loss Function

If you are given a model f(x) trained on the sample (x,y), the loss function B measures the absolute difference between the actual and current value. If these two last values have large vectoral distance, the loss function gives a high value. If the current and the actual values are close to each other, the loss function gives a very small value, close to 0. This is what all of the developers aim when constructing ANN or CNN models, to minimize the value of the loss function.

Some of the most used methods of the loss function are mean squared error (MSE), mean absolute error (MAE) etc.

2.4.1.4.3 Cost function

The objective of all of the machine learning and deep learning algorithm is to construct a model that will fulfill the requirement of predicting the outputs for specific inputs. This statement has been presented in a form of a hypothesis:

$$h_\theta (x) = \theta_0 + \theta_1 x$$

Theta represents the parameters that have to be found in order for the h(x) function to be as close as possible to the subset of the training data. The most suitable alternative would be to find out a function that will diminish the values of the parameters. The function that is used widely in this case is the mean square error, whose purpose is to calculate the difference between the actual data to the predicted data.

2.5 Convolutional Neural Networks

In this subsection, a brief presentation to Convolutional Neural Networks will be obtained. Deep learning, coupled with CNN form most of the current literature review in digital image processing which will be explained later on the Literature review chapter. A CNN model takes a dataset, preprocess it, and then it classifies in under categories that have been decided before starting the learning process. More specifically, firstly you have to standardize the input data to fit the model which may fine tuning or developed from scratch. The input size should be defined prior to feeding it into the model, but experiments on different input sizes can also be conducted in order to flourish the best accuracy out of it.

CNN architectures can accept both RGB and BW images. In our case, we have fed the model with RGB images and it takes them a 3-dimensional arrays, differently from the Black and White that is converted into a 2D array. After the step of input preparation finishes, the next one in line is the construction of convolutional layers.

2.5.1 Convolutional layer

Convolutional maintains the correlation among pixels by acquiring image characteristics applying small squares from input data. The convolutional layer implements a scientific method that accepts two inputs, the image matrix, and a filter or kernel. This is demonstrated on figure below:

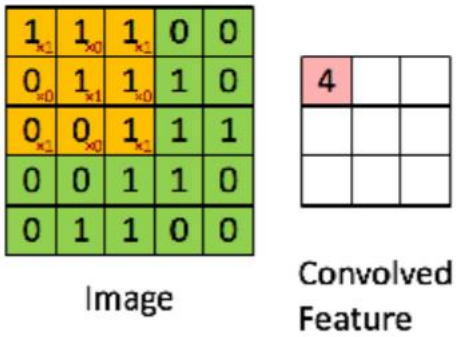

Figure 4 Convolution of the input signal (left) to convolved features(right). Per each pixel, convolution computes the dot product among input (green) and kernel (red). In this illustration, a 3x3 kernel has been applied.

2.5.2 Pooling layer

Pooling layers are employed from CNN for the dimensionality compression at images being too big, and removal of crucial characteristics, learning the important data. Spatial Pooling can be of different types: Max Pooling, Average Pooling, Sum Pooling

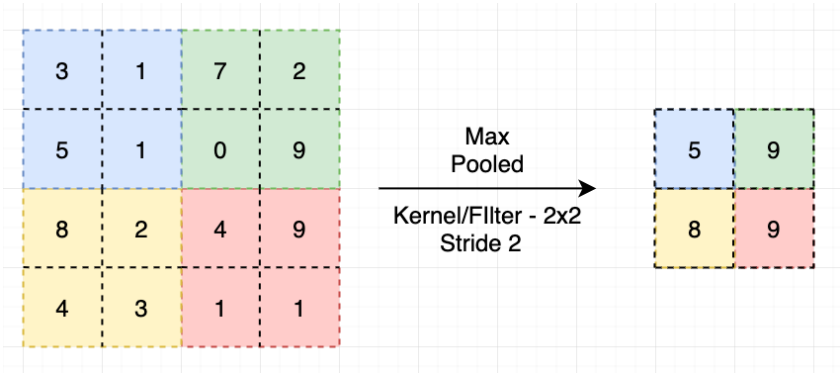

Figure 5 Max pooling of input matrix (left) to max pool layer (right). A 2x2 filter has been employed

2.5.3 Fully connected layer

The fully connected layers is the last layer of a CNN architecture. It usually utilized fully connected layers of neurons. Considering this, the matrix is flattened into a vector and then it is applied to a fully connected layer like a neural network. In this step all the features are consolidated together to shape a model. Lastly, an activation function is used to manage the object class.

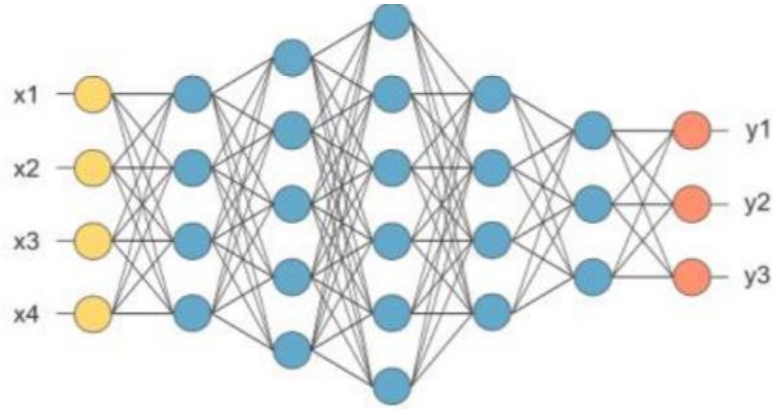

Figure 6 The matrix of the feature map is transformed into a vector and then is fed into the layers of the neural network

2.6 Ensemble Models

This subsection makes less part of the literature review. Only a few papers have been published on the usage of CNN as an ensemble learning on medical image analysis, especially on breast histopathological images. Ensemble learning is a technique, used in ANN models, in which multiple models are strategically combined in order to sustain improvements on the classification or prediction accuracy. The ensemble learning usually ensures higher accuracy in comparison to single trained models, because it tends to learn different features from the different models and has the ability to correct the errors of its members, acquiring in this way results that may have not be seen earlier.

There are some types of ensemble learning such as bagging, boosting, AdaBoost, and Stacked Generalization (Polikar). In this study, we are going to use the stacked generalization which uses two tiers of classifiers. The output of the first tier classifier serves as input for the second tier of the classifier. In this way, the system ensures if the data has been properly learned or not. This is presented in the following figure:

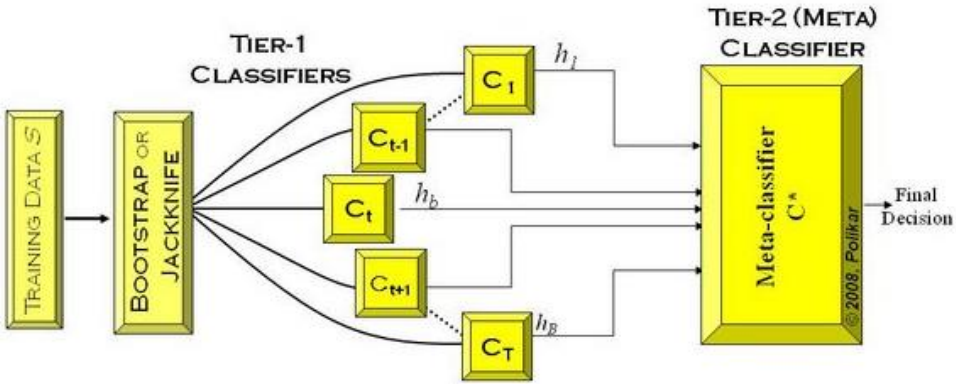

Figure 7 Stacked Generalization Schema

CHAPTER 3

LITERATURE REVIEW

3.1 Literature review

The development of Computer Aided Diagnosis system using deep learning, specifically Convolutional Neural Networks, can be utilized to help improve the accuracy of diagnosis. In the latest published papers, a lot of other methods have been employed for such operations, starting from hand crafted features, to machine learning algorithms, but the most used techniques which has achieved the best results on the literature, are the Convolutional Neural Networks architectures. Inspired by this indication, we have used two different fine-tuning CNN architectures (VGG16 and DenseNet201) whose result has been fed into a stacking ensemble learning model. Below, some other literature examples will be presented, further reasonably supporting our choice in this medical image study.

One study by Quoc Dang Vu suggests a system for the segmentation of nuclei and classification of tissue images by implementing a multiscale deep residual aggregation network to segment the cells and a classification algorithm that initially carries out patch-level classification via a deep learning method (Q.D.Vu, 2019). It achieves a 78% accuracy for the segmentation and 81% for the classification algorithm concluding that deep learning methods are improving in prediction performance and getting better at dealing with relatively small training datasets. Another study uses handcrafted-feature Convolutional Neural Network (CNN) used for binary and classification and AlexNet, U-net architectures for classifying mitosis in a histopathological tissue sample (G. Jiménez, 2019). It Achieves an accuracy of 95% for the Alexnet architecture and more than 95% for the U-net. For future research, this paper suggests using WSI directly because of the improvement it will have on the diagnosis of breast cancer as pathologists only evaluate small regions of the sample tissues. As conclusion, this paper states that deep learning approaches, combined with hand-crafted features, can noticeably improve the accuracy of the study.

Considering that the last paper mentioned have used a fine-tuning architecture for medical image analysis and performed very high accuracy, the same thing cannot be said for the conflict paper (J. Antony, 2016)and (E. Kim, 2016) had. In the previous study, fine-tuning certainly exceeded feature extraction, with a precision of 57.6% in classifying the grade of knee osteoarthritis while feature extraction accuracy being 53.4%. On the contrary, paper (E. Kim, 2016) revealed that feature extraction utilizing CNN gave better results than fine-tuning when classifying cytopathology (accuracy 70.5% vs 69.1%).A study made by Mittal (S.Mittal, 2019) implements 3 different CNN architectures for Stained Breast Tissue Images and inferred that among ResNet50, VGG19, and InceptionResNetV2, VGG19 arrived the highest accuracy of 96.14% and assured that it utilized the microenvironment and ensured that the results are consistent with underlying diagnoses. Another paper suggested DCNN for stain normalization and color argumentation and arrived at an F1 score of 0.82 by giving promising results on computational pathology (S. Otálora, 2019). A deep learning algorithm has also been suggested to be used by Shapcott too for the sampling in Colon Cancer Histology (M.Shapcott, 2019). It concluded that the usage of Within-image sampling improved performance without loss of accuracy.

Some other papers (Po-Hsuan (Cameron) Chen, 2018) (Y.Zhenga, 2017) (Yan Xu, 2015) (F. Xing, 2016) (Y.Xie, 2016) (Xie Y. , 2015) have used CNN algorithms for feature extraction and classification of images and have resulted in very promising accuracies. Recently, convolutional neural networks (CNNs) are becoming the standard for the classification challenges. A survey in deep learning for medical images reports that 36 out of 47 papers published on classification in 2015, 2016, and 2017 are using CNNs (G. Litjens, 2017).

As for the BreakHis dataset, the one that we are going to use in this study, some recent studies shows promising results. In (Duc My Vo, 2019), a compound of CNN and the boosting trees classifier were introduced for breast cancer classification and detection on BreakHis dataset. The suggested model applied Inception-ResNet-v2 model for visual feature extraction. Then a boosting classifier using gradient boosting trees was used for the final classification step. In (Chattoraj., 2019), an ensemble model was proposed for both binary and multi-class breast cancer detection on BreakHis dataset. Another aspect of the BreakHis dataset that one paper suggest has to do with the differences on the magnification factor (Benhammou, 2020). The paper

states that there are only a few papers in the literature (Bayramoglu, 2016) (Gupta, 2017), that uses the full dataset of BreakHis considering all of its magnification factors (40, 100, 200, 400X).

The eagerness to search for more ways that can help in increasing the classification accuracy, has made us find other helpful techniques. The usage of ensemble learning has led to promising results, containing many contest winning studies (Ensemble Methods: Elegant Techniques to Produce Improved Machine Learning Results., (2016, February 04).). Paper (Lu, 2017) suggests an ensemble learning model for breast MRI images by achieving a 0.96% accuracy. Similarly, paper (Mohebian, 2017) has employed a CAD for breast cancer recurrence using ensemble learning and achieved a 85% accuracy. In support of using ensemble learning is also paper (Wang, 2018) which has applied twelve different SVM for breast cancer diagnosis, having accuracy of 97%.

To sum up, there is said to have a lack of large training datasets and this has been mentioned as an obstacle of the image processing analysis. Also, the acquisition of relevant annotations or labeling for the medical images is thought to be a gap of this field. Furthermore, integrated approaches that combine deep/machine learning methods for analysis with deep/machine learning methods for quality assessment and data selection for iterative model refinement are said to provide an effective platform in pathology image analysis. (Xie J. L., 2019) CNN architectures seem to be the ones that have the highest performance in the literature, so the researchers are strongly encouraged to use them. Especially fine tuning methods are superior to the ones that are designed from scratch. On the other hand, the combination of different architectures by feeding them in ensembles, seems to be of a great interest for the past years.

For the aforementioned reasons, by taking into consideration what the literature suggests, in this study a stacking ensemble learning model for two fine tuning CNN architectures (VGG16 and Densenet201) will be implemented on the classification of breast tissues images using BreakHis dataset.

CHAPTER 4

EXPERIMENTS AND RESULTS

4. Experiments and Results

The state of the art studies have shown that there are different methodologies that can be used to detect, segment and classify the tissues in medical images, but what this study will be focused on is the usage of a fine-tuning ensemble architecture of CNN. Consequently, the methodology that will be used for this study will be deductive, meaning that the work will be based on already implemented architecture.

Another thing that is of great importance is choosing the right database that will be used for this research project. It is crucial to know that the image database that will be used, to be made of images with high quality in order to mitigate the chances of errors and increase the accuracy of the study.

After deciding of the CNN architectures and the database that this study will be based on, the next step is to start training the model. This model will be used to train the images of the dataset, by teaching them which one of the tissues is healthy and which one is not. After this phase is finished, the next one is to test the model and observe its accuracy.

Despite using the quantitative methodology for this study which included the implementation of the model that is explained above, it will also be based on qualitative methodology. This implies that the results of our model, the accuracy that it will give, will be compared to already implemented models and see which ones of them perform best in the classification of breast tissues. An overall qualitative analysis will be made considering the advantages and disadvantages of the study compared to the other ones.

4.1 Dataset Description

The dataset that has been used in this study has been published in 2016 by Spanhol et al. (Xie J. L., (2019, January 28)). It has a total of 7909 histopathological breast tissue images retrieved from 82 different patients. Surgical biopsies has been used by specialized doctors in order to transitive the original structure of the biological materials, providing in this way real life breast medical images. After this, the specialists collected via hematoxylin and eosin staining. Lastly, the output images were given labels, right after they were observed in high technology microscopes. The size of the images is 700x460 and they are RGB images. In the figure below, some examples of the images for both benign and malign can be seen:

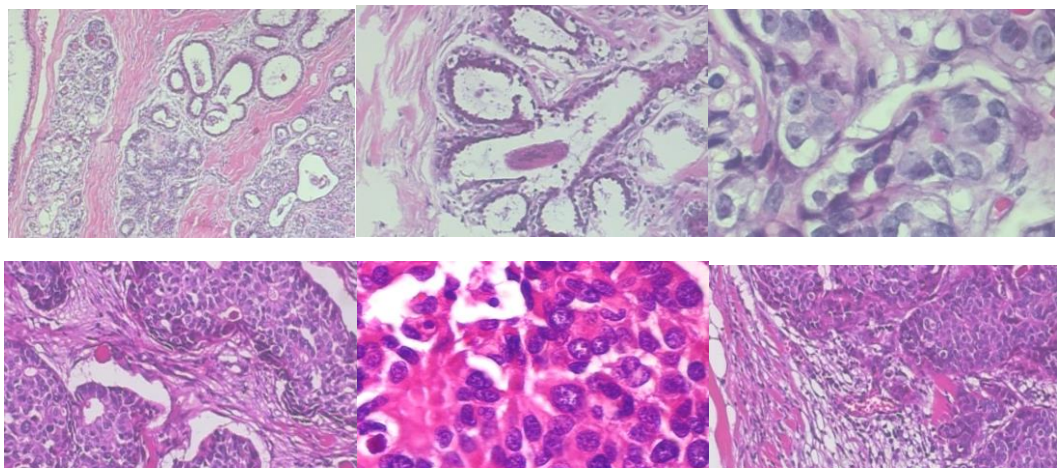

Figure 8 Benign breast tissue images (up), Malign breast tissue image (down)

The fact that there have been used different objective lenses, the database has been divided into 4 subsets containing images of magnification levels of 40x, 100x, 200x and 400x. These magnification subsets are each divided into malign and benign. Furthermore, the benign and malign are divided into 4 other subsets. Benign tumor images are separated into Adenosis (A), Fibroadenoma (F), Phyllodes Tumor (PT), and Tubular Adenoma (TA) while malign tumor images include Ductal Carcinoma (DC), Lobular Carcinoma (LC), Mucinous Carcinoma (MC), and Papillary Carcinoma (PC). Below a tabular representation of the data distribution is shown:

Table 1 The distribution of the dataset among its classes

Magnification	BENING				MALIGN				TOTAL
	A	F	PT	TA	DC	LC	MC	PC	
40X	114	253	109	149	864	156	205	145	1995
100X	113	260	121	150	903	170	222	142	2081
200X	111	264	108	140	896	163	196	135	2013
400X	106	237	115	130	788	137	169	138	1820
TOTAL	444	1014	453	569	3451	626	792	560	7909
PATIENTS	4	10	3	7	38	5	9	6	82

4.1.1 Dataset Preprocessing

In this thesis, we have adopted different preprocessing techniques such as data augmentation, image normalization which include the normalization of the pixel intensities and image size. In this section, we briefly explain each of them.

4.1.1.1 Data Augmentation

The limitation of the size of the medical images, specifically of the BreakHis dataset, causes overfitting of the data on the detection rate. What can mitigate this issue, is the usage of data augmentation. This technique aims to generate more training data from the current training set. There are several parameters that can be changed when using data augmentation, such as rotation, flipping, zooming etc. For Breakhis dataset, we have used the following parameters displayed on table below:

Table 2 Data Augmentation Parameters

Parameter	Value
Rotation Range	30
Width Shift Range	0.1
Height Shift Range	0.1
Shear Range	0.2
Zoom Range	0.2
Horizontal Flip	True
Fill Mode	Nearest

The following image shows how a breast tissue image changes when the augmentation parameters above are applied.

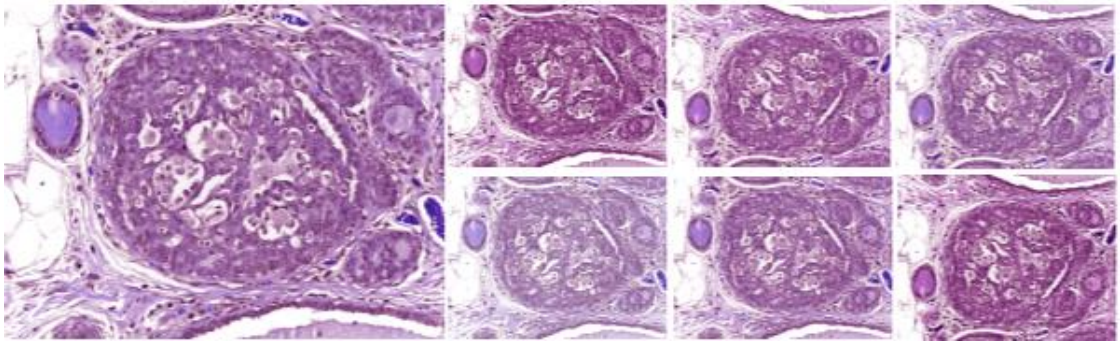

Figure 9 Images obtained after the application of data augmentation. (Left) the original image and (right) the images obtained.

4.1.1.2 Image Normalization

This step is also crucial when it comes to preprocessing a medical images dataset. The variability among people, differences in protocols between labs, fixation, specimen orientation in the block, human skills in tissue preparation, microscopy maintenance, and color variation due to differences in staining procedures, calls for the usage of image normalization (McCann MT, 2015). Its aim is to convey the same range of values of the images before it is fed to the CNN model. This also has an effect on the speed of the execution time. Firstly, the images have been resized to 224x224 from 700x460 that was the actual size to reduce the computational latency. And then a median filter has been applied in order to remove noise from the images. After this, scaling the range of raw pixel intensities, has helped in concluding the preprocessing step.

4.1.2 Dataset distribution for Testing and Training

Aster the preprocessing step finishes successfully, the division of the whole data set for training, testing and validation is part of the other step. As we have a total of 7909 breast tissues images, we have divided the training and testing split of 0.2. Before splitting these subsets, we have chosen a total of 150 breast images that have will ever be seen from the training or testing network, but will be used for validating the study.

A classification prediction accuracy will be calculated by using these 150 images from which 80 correspond to the malign class and 70 to the benign class. A tabular representation for the distribution of the BreakHis dataset can be observed below:

Table 3. The distribution of images among training testing and predicting for BreakHis dataset

	Total	Training(80%)	Validation(20%)	Predicting
Malign	5429	4279	1069	80
Benign	2480	1928	482	70
Total	7909	6207	1551	150

4.2 Network Architecture

This study demonstrates a successful application of deep learning architectures such as CNN, in the binary classification of breast histopathological images using stacking ensemble learning. We have used 2 different CNN architectures (VGG16 and DenseNet201) that are suggested by the literature to be very effective in breast cancer image classification. These two architectures have been selected by considering that they have committed successful applications in different computer vision tasks, being useful for real time applications such as our case, and finally because of being feasible with small datasets.

VGG16 (Zisserman., 2014) was launched by Karen Simonyan and Andrew Zisserman from Visual Geometry Group (VGG) of the University of Oxford in 2014. It is one of the highest performances in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014. The network employed 3×3 convolutional layers piled on top of every layer, shuffled with a max pooling layer, two 4096 nodes for fully-connected layers, and lastly supported by a softmax classifier.

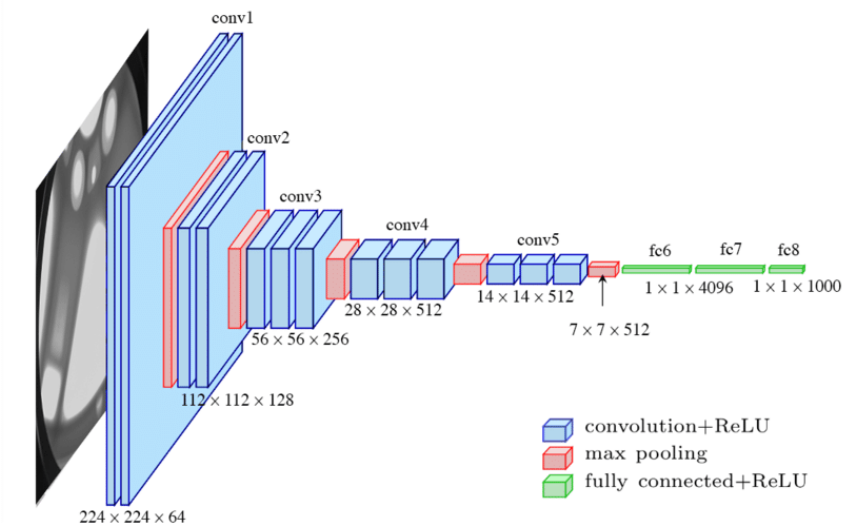

Figure 10 The architecture of VGG16

DenseNet201 (Gao Huang, 2017), stands for Densely-Connected Convolutional Networks, which is firstly introduced by Huang et al. (Gao Huang, 2017). DenseNet introduces a dense block, which is a sequential of convolutional layers, wherein every layer has a straightforward link to all consequent layers. This arrangement resolves the problem of vanishing gradient and increases feature distribution by applying really quick bonds connecting input and output layers everywherethenetwork

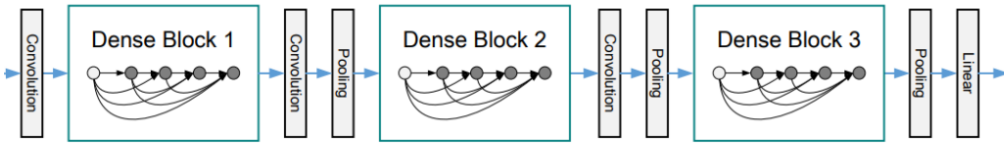

Figure 11 The architecture of DenseNet201

The technique used in this study is ensemble learning that is used in ANN models, in which multiple models are strategically combined in order to sustain improvements on the classification or prediction accuracy. The ensemble learning usually ensures higher accuracy in comparison to single trained models, because it tends to learn different features from the different models and has the ability to correct the errors of its members, acquiring in this way results that may have not be seen earlier.

In this study, we are going to use the stacked generalization which uses two tiers of classifiers. The output of the first-tier classifier serves as input for the second tier of the classifier. In this way, the system ensures if the data has been properly learned or not. More specifically, we train the fine tune model of VGG16 and DenseNet201 independently, save the models and then combine them together. The final fully connected layer of each of these models, are combined together to form a single feature vector. By combining the networks in this way, more information can be captured and a higher accuracy can be obtained. Finally, another multi-layered model has been used by having as an input the obtained feature vector fed into an MLP (Multi-Layer Perceptron)

4.3. Experiments and results

In this section the whole process of training and testing is described. As we have mentioned previously, the implementation will be based on fine tuning two existing CNN architectures, VGG16 and DenseNet201. The images of the dataset have been resized to 224x224 as the networks require.

Firstly the architectures have been loaded containing the pre-trained weights, and leave out the fully connected layers. In this way ,we can define a new fully connected layer on top of the base network. The next step is to freeze all the convolutional layers in the body of the network and continue training. In this phase only the head weight will be updated, as the other layers are frozen. This process is called warming up.

After finishing this phase, the next step is unfreezing the final CONV layer block. We start training the model again, but this time by including also the FC layer on the head of the network coupled with the final CONV block.

That said, to be more specific, we are going to explain the new layer each of the models have been added on their existing network. As mentioned earlier, it is a fully connected layer, with a ReLu activation function with 512 hidden neurons, followed by a dropout layer with a probability of 0.5 to prevent overfitting. The Dropout layer helps reducing over-fitting by randomly eliminating their contribution

on the training process. The SGD optimizer has been used with a learning rate of 1e-4 with a batch size set to 32 for the two models over 50 epochs. These were the hyperparameters used for the training process.

Both the networks presented above are build in Python and for training we used Keras having a Tensorflow backend. The operating system used is Windows 10, with an Intel(R) Core(TM) i7 -5600U 2.60 GHz , 16GB Ram, 256 SSD.

4.3.1 VGG16 Network

In this section the VGG16 network architecture will be described, together with the results it has offered to this study. But first let's explain in detail the form of the architecture. It has a total of 5 convolutional layers, which firstly accepts images of fixed size 224 x 224 RGB. Then the image is passed through a stack of convolutional layers, where the filters were used with a very small receptive field: 3×3. In one of the configurations, it also utilizes 1×1 convolution filters. The convolution stride is fixed to 1 pixel. Spatial pooling is carried out by five max-pooling layers, which follow some of the CONV layers. Max-pooling is performed over a 2×2 pixel window, with stride 2. Then three FC layers follow a stack of convolutional layers. The final layer is the soft-max layer.

As mentioned previously, the fine tuning method has been used in this study, meaning that the existing architecture has been modified by adding another fully connected layer. Firstly the FC layer has been removed from the model as it can be seen in the figure below.

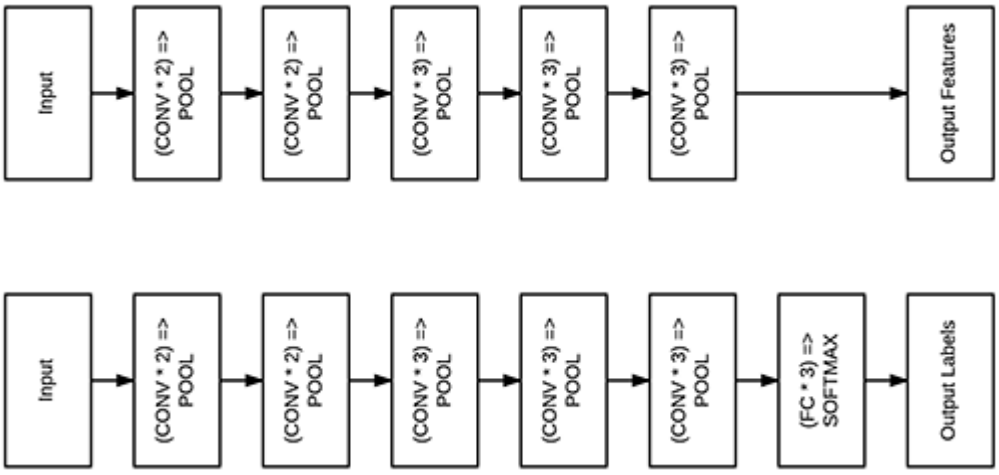

Figure 12 The original VGG16 architecture (down) and the modified fine tuning architecture, removing the last FC layer.

After this, another FC layer has been added at the top of the model as presented in the following figure :

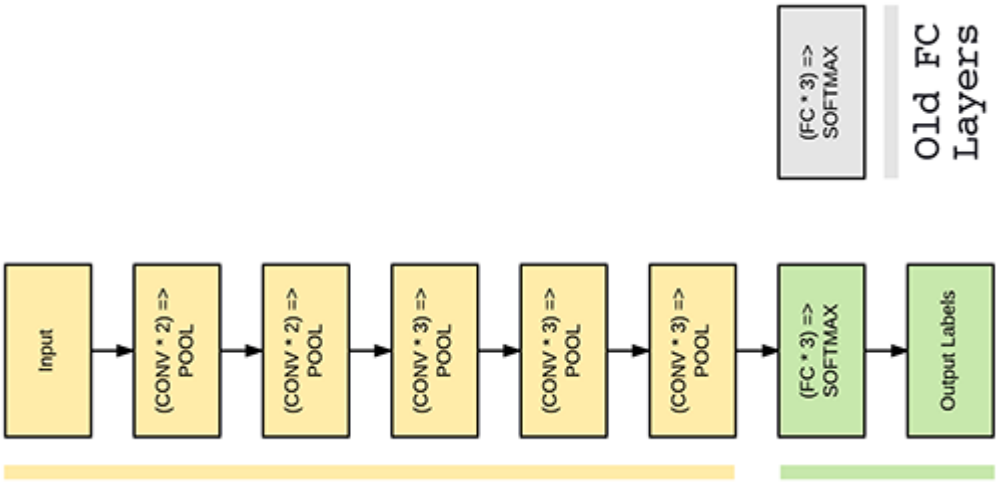

Figure 13 The existing layers of the VGG16 architecture(yellow), the new FC layer added to the network (Green), the old layer of the model(grey)

During the process of training that has been explained even in the previous section, The network firstly trains with frozen existing layers, activating only the FC added layer, then after finishing this process, the FC and CONV layers are trained. During these phases, we recorded the training and testing accuracy, as well as the loss.

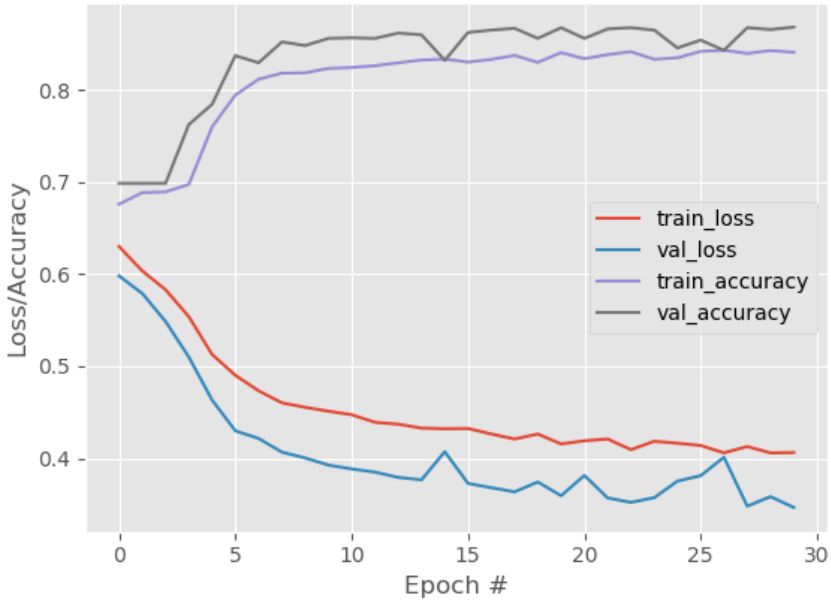

Figure 14 VGG16 unfreeze training and validation accuracy and loss

As it can be seen from the graph, we have terminated training at epoch 30 because of the training and accuracy loss started to divide. This happens in order to mitigate overfitting of the model. The model validation and training accuracy ranges from 70-88% and 68-87% respectively, conducting a good result. Additionally, this model received an accuracy of 86%.

4.3.2 DenseNet201 Network

In this section the DenseNet201 architecture will be described, coupled with the result that it has on this study. DenseNet is an CNN architecture that realizes the goal of easy training and parameter efficiency through feature reuse. The architecture of this network has been presented on the following figure.

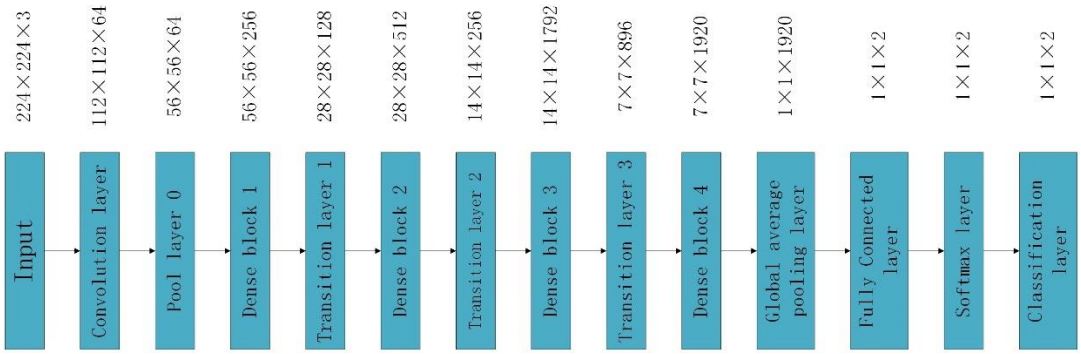

Figure 15 The architecture model of DenseNet201

The same thing happens with the DenseNet, when we have to modify it to achieve the fine tuning model. The last FC layer is removed and replaced with another one, the same as the one in VGG16. During the warm up and freezing procedure, we have measured the training and validation accuracy and loss. It is represented in the graph below:

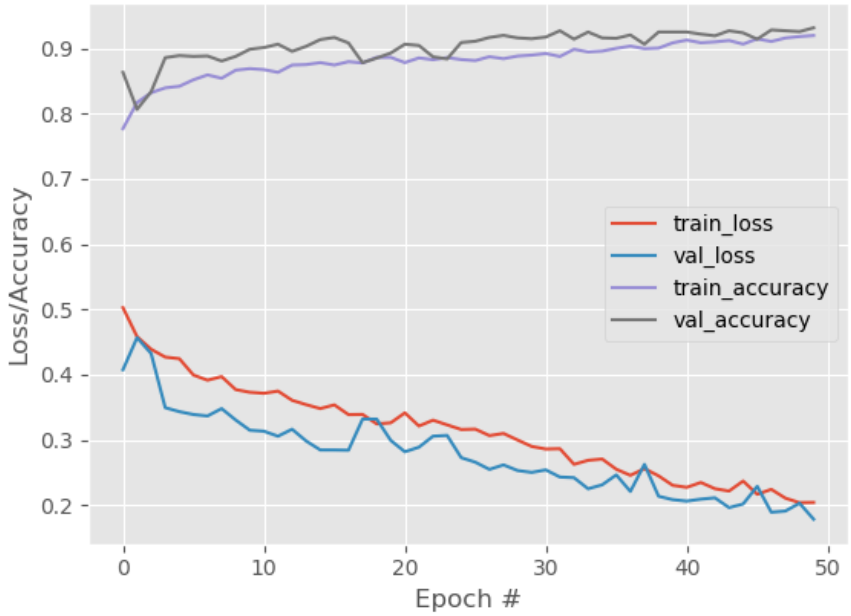

Figure 16 DenseNet201 Training and validation loss and accuracy during the unfreezing stage

As it can be seen from the graph, the range of accuracy is 79-92% for the training and 81-93%for the testing part. Finally, this model had a overall accuracy of 91%.

4.3.3 Ensemble Learning of two models

As it has been stated through all the parts of this study, the final stage of modeling the architectures, is feeding the results of both VGG16 and DenseNet201 into another model, Multilayer Perceptron model. The results of the prediction accuracy for this last part will be given in the following section.

4.4 Evaluate the model accuracy and compare the two models

To compare and evaluate both model we measured the accuracy during training phase. Two accuracies are reported in the previous section that are related to the training set and validation set. The validation accuracy better evaluates the ability of the model to generalize in new data because it is computed based on validation set with images that the model never sees during the training and it cannon memorize them. Other measurements taken in consideration are precision and recall. The table below summarizes all these measurements for both our models. From the results, it can be easily verified that the modified DenseNet201 outperforms the VGG16 architecture.

Table 4. Precision, recall, f1-score for VGG16 and DenseNet201

	Fine tuning VGG16	Fine Tuning DenseNet201
Precision	86	91
Recall	86	91
F1-score	86	91
Support	1551	1551
Best validation accuracy	87	93
Worst validation accuracy	70	81
Average validation accuracy	79	87

After training and testing both of the above architectures, we have to make the predictions for each of them, and see how many images that have never be seen from the network, will be predicted. As stated previously, a total of 150 images have been selected from the database where 80 are malign images and 70 benign.

We have tested both the architectures separately, and received a prediction accuracy of 89.05% for VGG and 90.02% for the DenseNet201. It means that the

VGG16 has predicted 133 correct images out of 150 and the DenseNet has predicted 135 correct images.

Finally, the overall ensemble model has been tested in order to obtain the prediction accuracy. It arrived to predict 136 images out of 150 and achieved 91.03% accuracy. Hence, this ensemble model achieves a higher prediction accuracy in comparison to the independent architectures of VGG16 and DenseNet201.

CHAPTER 5

CONCLUSIONS AND FUTURE WORK

5. Conclusion and Future Work

The focus of this thesis is to develop and generalize a method on classifying breast histopathological images into healthy and not healthy by utilizing a stacking ensemble method. From the literature review, we concluded that CNN architectures, coupled with fine tuning fed into an ensemble learning model, outperformed the other traditional image processing techniques. Previous work in classifying breast tissues have shown promising results, but very few have incorporated both fine tuning and ensemble learning models.

The proposed method in this thesis is fine tuning, which modify an existing architecture of CNN. We have modified both VGG16 and DenseNet201, and received a prediction accuracy of 89.05% and 90.02% respectively. When these two models have ben fed into an ensemble architecture, undergoing another training phase by utilizing MLP architecture, the prediction accuracy improved into 91.03%.

The results are promising, but if more architectures would be fed into the ensemble model, some other features might have been learned by the model by giving in this way an improvement in the accuracy. Also, the ensemble models can be the foundation to the integration of deep learning and blockchain technology, since it preserves the anonymousness and the privacy of patient data.

References

WHO-Breast cancer (n.d.). Retrieved from https://www.who.int/cancer/prevention/diagnosisscreening/breast-cancer/en/

Bayramoglu, N. K. (2016). Deep learning for magnification independent breast cancer histopathology image classification. *23rd International Conference on Pattern Recognition (ICPR). doi:10.1109/icpr.2016.7900002*.

Benhammou, Y. A. (2020). BreakHis based breast cancer automatic diagnosis using deep learning: Taxonomy, survey and insights. *Neurocomputing, 375,* . *doi:10.1016/j.neucom.2019.09.044*, 9-24.

C. E. Nwankpa, W. I. (2018). Activation Functions: Comparison of Trends in Practice and Research for Deep Learning. *ArXiv*.

Chattoraj., S. P. (2019). Diving Deep onto Discriminative Ensemble of Histological Hashing & Class-Specific Manifold Learning for Multi-class Breast Carcinoma Taxonomy. *ICASSP 2019-2019 IEEE International Conference on Acoustics*.

Chu, S. M. (2019). Ensemble deep learning-based fault diagnosis of rotor bearing systems. . *Computers in Industry 105 (2019), https: //doi.org/10.1016/j.compind.2018.12.012*, 143 – 152.

Duc My Vo, N.-Q. N.-W. (2019). Classification of breast cancer histology images using incremental boosting convolution networks. *Information Sciences 482 https://doi.org/10.1016/j.ins.2018.12. 089*, 123 – 138.

E. Kim, M. C.-R. (2016). A deep semantic mobile application for thyroid cytopathology. *Medical Imaging 2016: PACS and Imaging Informatics: Next Generation and Innovations*.

Ensemble Methods: Elegant Techniques to Produce Improved Machine Learning Results. ((2016, February 04).). *https://www.toptal.com/machine-learning/ensemble-methods-machine-learning*.

F. Xing. (2016). Transfer Shape Modeling Towards High-Throughput Microscopy Image Segmentation.

G. Jiménez. (2019). Deep Learning for Semantic Segmentation vs. Classification in Computational Pathology: Application to Mitosis Analysis in Breast Cancer Grading.

G. Litjens. (2017). A survey on deep learning in medical image analysis.

Gao Huang, Z. L. (2017). Densely connected convolutional networks. *In Proceedings of the IEEE conference on computer vision and pattern recognition*, 4700–4708.

Gupta, V. &. (2017). Breast Cancer Histopathological Image Classification: Is Magnification Important? .

J. Antony, K. M. (2016). Quantifyi,ng radiographic knee osteoarthritis severity using deep convolutional neural networks. *Arxiv: 1609.02469*.

Lu, W. L. (2017). A novel computer-aided diagnosis system for breast MRI based on feature selection and ensemble learning. *Computers in Biology and Medicine, 83, . doi:10.1016/j.compbiomed.2017.03.002* , 157-165.

M.Shapcott, ". (2019). Deep Learning With Sampling in Colon Cancer Histology.

Matas, D. J. (2016). All you need is a good init. *ArXiv*.

McCann MT, O. J. (2015). Automated histology analysis: Opportunities for signal processing. *IEEE Signal Processing Magazine. 2015;32(1):78*.

Mohebian, M. R. (2017). A Hybrid Computer-aided-diagnosis System for Prediction of Breast Cancer Recurrence (HPBCR) Using Optimized Ensemble Learning. Computational and Structural Biotechnolo.

Mostafa Amin-Naji, A. A. (2019). Ensemble of CNN for multi-focus image fusion. *Information Fusion 51 https://doi.org/10.1016/j.inffus.2019.02.003,* 201 – 214.

Oscar Perdomo, H. R. (2019). Classification of diabetes-related retinal diseases using a deep learning approach in optical coherence tomography.

P. Ramachandran, B. Z. (2017). Searching for Activation Functions. *ArXiv,*.

Po-Hsuan (Cameron) Chen. (2018). Microscope 2.0: An Augmented Reality Microscope with Real-time Artificial Intelligence Integration.

Polikar, R. (. (n.d.). Ensemble learning. *http://www.scholarpedia.org/article/Ensemble_learning#:~:text=Ensemble learning is the process,, function approximation, etc.*

Q.D.Vu. (2019). Methods for Segmentation and Classification of Digital Microscopy Tissue Images.

S. Otálora. (2019). Staining Invariant Features for Improving Generalization of Deep Convolutional Neural Networks in Computational Pathology.

S.Mittal. (2019). Digital Assessment of Stained Breast Tissue Images for Comprehensive Tumor and Microenvironment Analysis.

Wang, H. Z. (2018). A support vector machine-based ensemble algorithm for breast cancer diagnosis. *European Journal of Operational Research, 267(2),. doi:10.1016/j.ejor.2017.12.001* , 687-699.

Xie, J. L. ((2019, January 28)). Deep Learning Based Analysis of Histopathological Images of Breast Cancer. *Retrieved from https://www.frontiersin.org/articles/10.3389/fgene.2019.00080/full#B34*.

Xie, J. L. (2019). Deep Learning Based Analysis of Histopathological Images of Breast Cancer. *Frontiers in Genetics, 10. doi:10.3389/fgene.2019.00080*.

Xie, Y. (2015). Deep Voting: A Robust Approach Toward Nucleus Localization in Microscopy Images.

Y.Xie. (2016). Spatial Clockwork Recurrent Neural Network for Muscle Perimysium Segmentation.

Y.Zhenga. (2017). Feature extraction from histopathological images based on nucleus-guided convolutional neural network for breast lesion classification.

Yan Xu. (2015). DEEP CONVOLUTIONAL ACTIVATION FEATURES FOR LARGE SCALE BRAIN TUMOR HISTOPATHOLOGY IMAGE CLASSIFICATION AND SEGMENTATION.

Zisserman., K. S. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556 (2014)*.

Appendix

Appendix A – Training and Testing Code

```
import matplotlib

matplotlib.use("Agg")

# import the necessary packages
from keras.preprocessing.image import ImageDataGenerator
from keras.applications import VGG16, DenseNet201
from keras.layers.core import Dropout
from keras.layers.core import Flatten
from keras.layers.core import Dense
from keras.layers import Input
from keras.models import Model
from keras.optimizers import SGD, Adam
from sklearn.metrics import classification_report
from pyimagesearch import config
from imutils import paths
import matplotlib.pyplot as plt
import numpy as np
import pickle
import os


def plot_training(H, N, plotPath):
    # construct a plot that plots and saves the training history
    plt.style.use("ggplot")
    plt.figure()
    plt.plot(np.arange(0, N), H.history["loss"], label="train_loss")
    plt.plot(np.arange(0, N), H.history["val_loss"], label="val_loss")
    plt.plot(np.arange(0, N), H.history["accuracy"], label="train_acc")
    plt.plot(np.arange(0, N), H.history["val_accuracy"], label="val_acc")
    plt.title("Training Loss and Accuracy")
    plt.xlabel("Epoch #")
    plt.ylabel("Loss/Accuracy")
    plt.legend(loc="lower left")
    plt.savefig(plotPath)


# derive the paths to the training, validation, and testing
# directories
trainPath = os.path.sep.join([config.BASE_PATH, config.TRAIN])
valPath = os.path.sep.join([config.BASE_PATH, config.VAL])
testPath = os.path.sep.join([config.BASE_PATH, config.TEST])
```

```
# determine the total number of image paths in training, validation,
# and testing directories
totalTrain = len(list(paths.list_images(trainPath)))
totalVal = len(list(paths.list_images(valPath)))
totalTest = len(list(paths.list_images(testPath)))

# initialize the training data augmentation object
trainAug = ImageDataGenerator(
  rotation_range=30,
   zoom_range=0.15,
   width_shift_range=0.2,
   height_shift_range=0.2,
   shear_range=0.15,
   horizontal_flip=True,
   fill_mode="nearest")

# initialize the validation/testing data augmentation object (which
# we'll be adding mean subtraction to)
valAug = ImageDataGenerator()

# define the ImageNet mean subtraction (in RGB order) and set the
# the mean subtraction value for each of the data augmentation
# objects
mean = np.array([123.68, 116.779, 103.939], dtype="float32")
trainAug.mean = mean
valAug.mean = mean

# initialize the training generator
trainGen = trainAug.flow_from_directory(
   trainPath,
   class_mode="categorical",
   target_size=(224, 224),
   color_mode="rgb",
   shuffle=True,
   batch_size=config.BATCH_SIZE)

# initialize the validation generator
valGen = valAug.flow_from_directory(
   valPath,
   class_mode="categorical",
   target_size=(224, 224),
   color_mode="rgb",
   shuffle=True,
   batch_size=config.BATCH_SIZE)

# initialize the testing generator
testGen = valAug.flow_from_directory(
   testPath,
   class_mode="categorical",
   target_size=(224, 224),
   color_mode="rgb",
   shuffle=True,
   batch_size=config.BATCH_SIZE)
```

```
# load the Densenet201 network, ensuring the head FC layer sets are left
# off
baseModel = DenseNet201(weights="imagenet", include_top=False,
                  input_tensor=Input(shape=(224, 224, 3)), pooling=None, classes=2)

# construct the head of the model that will be placed on top of the
# the base model
headModel = baseModel.output
headModel = Flatten(name="flatten")(headModel)
headModel = Dense(512, activation="relu")(headModel)
headModel = Dropout(0.5)(headModel)
headModel = Dense(len(config.CLASSES), activation="softmax")(headModel)

# place the head FC model on top of the base model (this will become
# the actual model we will train)
model = Model(inputs=baseModel.input, outputs=headModel)

# loop over all layers in the base model and freeze them so they will
# *not* be updated during the first training process
for layer in baseModel.layers:
   layer.trainable = False

# compile our model (this needs to be done after our setting our
# layers to being non-trainable
print("[INFO] compiling model...")
opt = Adam(lr=0.0001)
model.compile(loss="categorical_crossentropy", optimizer=opt,
          metrics=["accuracy"])

# Randomize the data generated

# train the head of the network for a few epochs (all other layers
# are frozen) -- this will allow the new FC layers to start to become
# initialized with actual "learned" values versus pure random
print("[INFO] training head...")
H = model.fit_generator(
   trainGen,
   steps_per_epoch=totalTrain // config.BATCH_SIZE,
   validation_data=valGen,
   validation_steps=totalVal // config.BATCH_SIZE,
   epochs=50)

# reset the testing generator and evaluate the network after
# fine-tuning just the network head
print("[INFO] evaluating after fine-tuning network head...")
testGen.reset()
predIdxs = model.predict_generator(testGen,
                        steps=(totalTest // config.BATCH_SIZE) + 1)
predIdxs = np.argmax(predIdxs, axis=1)
print(classification_report(testGen.classes, predIdxs,
                     target_names=testGen.class_indices.keys()))
plot_training(H, 50, config.WARMUP_PLOT_PATH)

# reset our data generators
trainGen.reset()
```

```
valGen.reset()

# now that the head FC layers have been trained/initialized, lets
# unfreeze the final set of CONV layers and make them trainable
for layer in baseModel.layers[15:]:
    layer.trainable = True

# loop over the layers in the model and show which ones are trainable
# or not
for layer in baseModel.layers:
    print("{}: {}".format(layer, layer.trainable))

# for the changes to the model to take affect we need to recompile
# the model, this time using SGD with a *very* small learning rate
print("[INFO] re-compiling model...")
opt = Adam(lr=0.0001)
model.compile(loss="categorical_crossentropy", optimizer=opt,
            metrics=["accuracy"])

# train the model again, this time fine-tuning *both* the final set
# of CONV layers along with our set of FC layers
H = model.fit_generator(
    trainGen,
    steps_per_epoch=totalTrain // config.BATCH_SIZE,
    validation_data=valGen,
    validation_steps=totalVal // config.BATCH_SIZE,
    epochs=50)

# reset the testing generator and then use our trained model to
# make predictions on the data
print("[INFO] evaluating after fine-tuning network...")
testGen.reset()
predIdxs = model.predict_generator(testGen,
                            steps=(totalTest // config.BATCH_SIZE) + 1)
predIdxs = np.argmax(predIdxs, axis=1)
print(classification_report(testGen.classes, predIdxs,
                        target_names=testGen.class_indices.keys()))
plot_training(H, 50, config.UNFROZEN_PLOT_PATH)

# serialize the model to disk
print("[INFO] serializing network...")
model.save(config.MODEL_PATH)
```

Appendix B – Predicion code

```
# import the necessary packages
from keras.models import load_model
from pyimagesearch import config
import numpy as np
import argparse
import imutils
import cv2
```

```
# construct the argument parser and parse the arguments
ap = argparse.ArgumentParser()
ap.add_argument("-i", "--image", type=str, required=True,
  help="path to our input image")
args = vars(ap.parse_args())

# load the input image and then clone it so we can draw on it later
image = cv2.imread(args["image"])
output = image.copy()
output = imutils.resize(output, width=400)

# our model was trained on RGB ordered images but OpenCV represents
# images in BGR order, so swap the channels, and then resize to
# 224x224 (the input dimensions for VGG16)
image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
image = cv2.resize(image, (224, 224))

# convert the image to a floating point data type and perform mean
# subtraction
image = image.astype("float32")
mean = np.array([123.68, 116.779, 103.939][::1], dtype="float32")
image -= mean

# load the trained model from disk
print("[INFO] loading model...")
model = load_model(config.MODEL_PATH)

# pass the image through the network to obtain our predictions
preds = model.predict(np.expand_dims(image, axis=0))[0]
i = np.argmax(preds)
label = config.CLASSES[i]

# draw the prediction on the output image
text = "{}: {:.2f}%".format(label, preds[i] * 100)
cv2.putText(output, text, (3, 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5,
  (0, 255, 0), 2)

# show the output image
cv2.imshow("Output", output)
cv2.waitKey(0)
```