Histopathological Image Classification using Deep Learning
by
Braulio Arredondo Padilla

Abstract

This thesis presents a study of digital pathology classification using and combining several techniques of machine learning and deep learning. Cancer is one of the most common causes of death around the world. One of the main complications of the disease is the prediction in the final stage. Nowadays there are many different studies to obtain a correct diagnosis on time. Some of these studies are tissue biopsies. These samples are analyzed by a pathologist, which must observe pixel by pixel a whole image of high dimensions to give a diagnostic of the disease, including stage and class. This activity takes weeks, even for experts, because usually several samples are extracted from a single patient. To speed up and facilitate this process, several models have been developed for digital pathology classification. With these models, it is easier to discard many patient slides than the traditional method, then, the main activity for a pathologist is to confirm a diagnosis with the most relevant or complicated sample. The downside of these models is that most of them are based on deep learning, a technique that is well known for its great performance, but also for its high requirements like graphic processors and memory resources. Consequently, we performed a complete analysis of several convolutional neural networks used in different ways to compare outcomes and efficiency. In addition, we include techniques such as recurrent neural networks and machine learning. Several models of deep learning and machine learning are presented as alternatives to convolutional neural networks, including 5 computer vision techniques. The main objective of our project is to perform a real alternative capable to achieve similar outcomes to deep learning with limited resources. The experiments were successful, including a real alternative for deep learning for the classification of 3 different types of cancer with an area under the curve higher than 90%.

Chapter 1

Introduction

According to the World Health Organization, cancer is the second cause of death in the world. There are estimated 9.6 million deaths in 2018. From these deaths, 70% are from low-middle income countries. A third part of the cancer deaths comes from 5 principal risks, which are the consume of tobacco and alcohol, high body mass, low consume of vegetables and fruits and low psychical activity. Some of these risks are more related with colon and laugh cancer [1]. There are some other cancer types like breast cancer, where each year, 1.38 millions of new cases are detected around the world and 458,000 deaths are reported by this disease.

For cancer detection, have been developed several techniques, like DNA and RNA sequencing [2]. These techniques can obtain complete information from the cancer disease, including type, subtype and phase. The main inconvenient of sequencing is the technology required and the high price of the process, which means that is not easy to get access to these studies at many medical centers, at least, in the middle-income countries. There are other techniques that are more common and necessaries to confirm a diagnostic, like tissue extraction from tumors. These interventions are more popular for some types of cancer like breast cancer where there is a high diversity over tissue classes and specific treatments for each one. The tissue sample extracted from the tumor is analyzed by a pathologist, the expert marks the areas of interest; where cells, glands, necrosis, and other elements are presented. According to this analysis, the tissue is classified as benign or malignant, in some cases a specific benign or malignant tissue subtype is diagnosed for a corresponding treatment. Tissue classification is a task that can be performed on several days, even weeks, for an expert. Generally, several samples are extracted from a patient, the samples could be represented as a digital image of more than 100,000 x 100,000 pixels which has to be analyzed pixel by pixel by the pathologist [3].

Nowadays, to support pathologist on this task, have been developed several models for the classification of histopathological images. Deep learning and machine learning are the most popular approaches to tackle the problem. In the last decade, several convolutional network architectures have been published. Year by year, a new architecture is presented with better performance than the previous one on the most complex problems like ImageNet [7]. Something else that increases year by year is the float point operations per network, increasing

the hardware requirements to train it; requirements that are not cheap and are not accessible for everyone. But fortunately, these models are not the only ones capable to tackle this task. Due to the constant creation of models for the complex problems like ImageNet, transfer learning and fine tuning are some strategies to use convolutional networks without needing to train a complete model from scratch. Unfortunately, even when these techniques work for some of the common problems related with objects and creatures, are not enough to achieve good results in comparison with full trained models.

Until this point, it is possible to conclude that, without the hardware resources is not possible to create a model capable to tackle a task like the digital pathology; but there is always a final option. Image processing is still being a viable technique capable to achieve great results if it is compared with other techniques like deep learning. Handcrafted features are not obtained from pretrained models, these algorithms just read pixel features according to a mathematical equation designed to calculate borders, patterns, histograms and more. Several projects have compared different handcrafted features with other features extracted from convolutional neural networks [4]. But even when the handcrafted features and image processing work good and they do not need high resources like graphic processors or memory, they are still being less popular than deep learning techniques for this task.

In digital pathology, computer vision techniques are performed with an almost insignificant proportion compared with deep learning experiments. For example, Bardou et al.compares SURF and Dense SIFT features with features extracted by convolution networks [5]. Sharma et al. repeat the experiment but using Haralick textures, Hu moment and color histogram [6]. Even when these experiments cover the same area of interest, collection of handcrafted features is increasing, and these experiments are just a small part of all the possible combinations.

Extending the use of handcrafted features on digital pathology datasets is important to support the use of different techniques, and not only apply deep learning. Even when these features are not used frequently for complex problems like pathology. These techniques can achieve an amazing and complete review of each image, without read a complete dataset to get the main characteristics from a class.

1.1 Problem Statement

Digital pathology classification, different from common image problems, requires some specific considerations. Histopathological images are the digitization of tissue samples extracted from tumors. The objective of these images is to determine the tumor subtype, obtained by an exhaustive analysis of the pathologist. The main classification is if a tumor is benignant or malignant, and then which is the subtype of the sample and if there is an available specific treatment. The process could sounds simple, but the fact is that that images could have more than 10,000,000,000 pixels, and the sample sometimes needs an extremely detailed analysis that includes read the whole image using windows from 256x256 to 512x512 pixels, justifying in some cases, why many weeks are expended for the pathology diagnosis[3].

Parting from this problem, intelligent solutions are trained just with the pathologist annotation from the tissue class desired to classify. We must clarify that not all the tissue sample contains cancer indications, and there are others which contains more than one subtype. There are other approaches that do not require pathologist annotations; but require higher and considerable hardware resources, like weakly-supervised learning using whole slide images [3].

Even with the pathologist annotations, which reduces in high factor the possible number of images for training, is not easy to train a model. There are other important factors to consider than the number of images, for example the dimension of each image comes from 512 to 2000 pixels in some datasets of annotated slides. Even for convolution networks, these dimensions still being big for the models. Information on the image is a factor that could improve the results, for example filter the images with high borders of background is a good technique.

Usually these are just situations presented on other problems. But, in pathology there are factors out of common. The color of slides according of the quality and quantity of eosin and hematoxylin diverge a lot from each patient and each medical center. The scanner resolution has a big influence over the image quality and resolution, which also is different between each medical center. Finally, the augmentation of the image must be measured carefully, with high augmentation we can loss characteristics like the form, location and environment of cells and glands. Conversely, with a low augmentation, we can loss specific details like cells, nucleus and texture patterns. These are just some of the drawbacks that must be considered for a possible solution.

Finally, it must be clarified that even with the best model reported, no one of these solutions has been used to determine a final diagnostic of a patient. Solutions are used for pathologist support, and some others to sort the image from the most complicated samples, with not conclusive results, to the easiest samples, with the highest predictions close to 1 for any class. Doing this, an expert can expend more time for the complex images and confirming the diagnostic with the last images of the patient, avoiding the hole analysis of each patient image. Generally, there are extracted around 5 samples from a patient tissue, and in some cases, cancer clues could not be found in all of them.

1.2 Objectives

The general objective of the project is to generate, evaluate and compare different models capable of classify different histopathological images of different types. The models need to be designed focusing on different techniques of deep learning, machine learning and image processing. The performance of each model will be done using main metrics like accuracy and the area under the curve of the receiver operating characteristic. To achieve this general objective, the following objectives are considered:

- Implement at least 4 different variations of convolutional neural networks.

- Implement at least 10 different machine learning techniques.
- Obtain features from 5 different handcrafted features.
- Test the models over 3 different types of cancer.
- Compare results between deep learning features and handcrafted features over machine learning techniques.
- Compare the results obtained with the results presented at literature.

1.3 Research Questions

Implementing different variations of deep learning and feature extraction like transfer learning and fine tuning will generate results with many combinations due the possible convolution network architectures and machine learning techniques. Similar for the combination of handcrafted features and machine learning, they will give us many possible results, allowing us to answer the following questions.

- Have the models similar results for different types of cancer?
- Are transfer learning and fine-tuning viable options for hardware requirements limitation?
- Have technical issues like eosin and hematoxylin coloration, digital scanning and image augmentation considerable influence over classification results?
- Have machine learning techniques similar results over the same image features?
- Are handcrafted features capable to replicate the results obtained from deep learning techniques?

1.4 Solution Overview

To carry out this project successfully, the workflow has been divided into 4 categories: Data, Deep Learning, Machine Learning and Handcrafted Features. For the input of these models, different datasets were collected, examined, and processed according to characteristics like patient distribution, augmentation factors, number of classes, sub-classes, coloration and other factors. Deep learning extends different implementations like image pre-processing, features extraction and fine tuning over different architectures of convolution networks and recurrent networks. Once extracted the features, 10 machine learning techniques are implemented to calculate a final classification using the extracted features from deep learning. Finally, using image processing, all the handcrafted features are used to train the machine learning techniques, also combination and comparison of these features is reported.

Chapter 2

Data

For the project, two different datasets were used. One of the objectives is trying to extend the research to distinct techniques but also to distinct patients and cancer types to. One of the main inconveniences that is found on digital pathology is the data format. The Cancer Gnome Atlas is probably the principal data portal on the cancer research. Specifically, for pathology there are thousands and thousands of whole slide images for the more than 30 different cancer types. Unfortunately, these slides do not have the pathologist annotation. This means that we only have the cancer class of the whole slide, but not the specific areas where is the cancer. In that specific scenario the only possible solution is the weakly-supervised learning, but it requires high hardware resources to achieve the task of manage thousands and thousands of whole slide images[3].

Other datasets are the annotated, which include the specific areas with cancer presence. These annotations make it easier the management of images, smaller size, less number of elements, and more concentration of information. We decide to use these kinds of dataset due the desired multiple experiments and combinations. Also, is necessary to clarify that public histopathological dataset are not easy to obtain due the patient's privacy, space required, and pathologist work involved. The selected datasets are the following.

2.1 LC25000 Dataset

Lung and Colon Cancer Histopathological Image Dataset is released at the end of 2019 [8]. This dataset is compounded by 25,000 as is presented in table 2.1 images of 2 cancer types, lung and colon. Colon cancer images contains 5,000 images of colon adenocarcinoma and 5,000 images of benign colonic tissue. Lung cancer images contains 5,000 images of lung adenocarcinoma, 5,000 images of lung squamous cell carcinoma and 5,000 images of benign

Table 2.1: LC25000 Dataset distribution

	Colon		**Lung**		
	Adenocarcinoma	**Benign**	**Adenocarcinoma**	**Squamous**	**Benign**
Total of Images	5,000	5,000	5,000	5,000	5,000

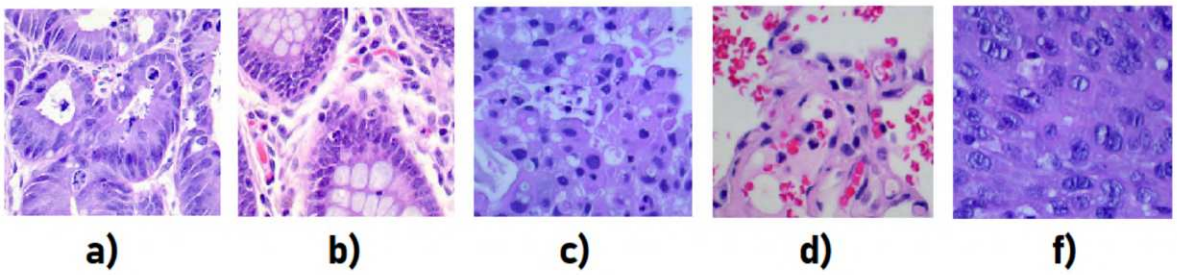

Figure 2.1: a) Colon Adenocarcinoma, b) Colon Benign, c) Lung Adenocarcinoma d) Lung Benign e) Lung Squamous

lung tissue.

To generate the dataset, 750 images of lung tissue were used, 250 benign lung tissue, 250 lung adenocarcinomas, and 250 lung squamous cell carcinomas. A total of 500 images of colon tissue were used, 250 of benign colon tissue and 250 of colon adenocarcinomas. All of them were captured from pathology glass slides. All images were cropped to square sizes of 768 x 768 pixels from original 1024 x 768 pixels and saved on jpe format. This is a pre-processed dataset. The images were augmented using Augmentor library in Python [9]. Using the same library, the 25,000 images were obtained by applying left and right rotations up to 25 degrees and a probability of 1. Horizontal and vertical flips were applied with 0.5 probability. For the experimentation, the data splitting is done by 4,000 images per class for training and 1,000 per class for testing.

2.2 BreakHis Dataset

Breast Cancer Histopathological Database by Spanhol et al. was published in 2017 [10]. The database was built in collaboration with the P&D Laboratory - Pathological Anatomy and Cytopathology, this dataset is composed by 9,109 images breast tumor slide images. These images were collected from 82 patients. The data is compounded by different magnification factors, 40X, 100X, 200X, and 400X, these factors are presented at figure 2.2. There are in total 2,480 benign images and 5,429 malignant images as we can see in the distribution of the table 2.2. The images have dimensions of 700 X 460 pixels in a png format.

BreaKHis is compounded of 2 main classes, benignant and malignant. The presented samples were collected by SOB method, also known as partial mastectomy or biopsy. The procedure compared to any methods of needle biopsy, removes the larger size of tissue and is done with general anesthetic. Both classes can be separated into different types by the tumoral cells at microscopical appearance. Some of these subtypes can have different prognoses, diagnostics and treatments.

The benignant class contains the following tumor types:

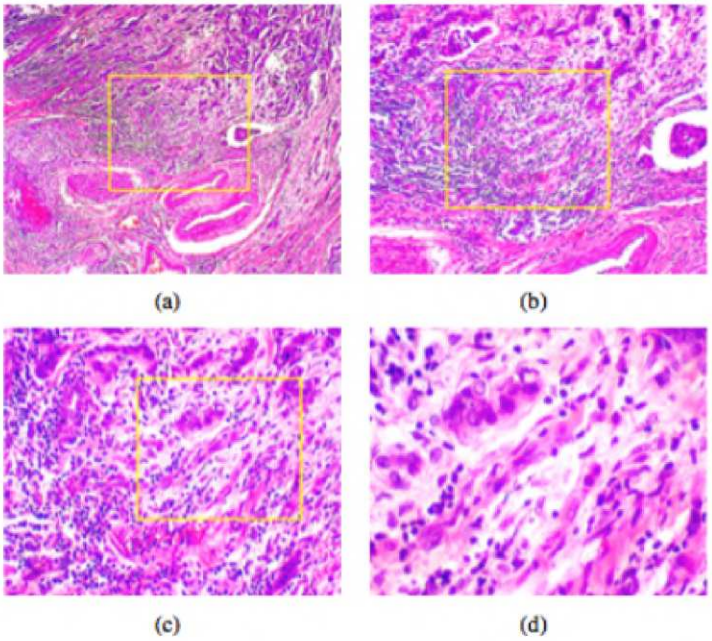

Figure 2.2: BreakHis Dataset slide magnification example by Spanhol et al. [10]. (a) 40X, (b) 100X, (c) 200X, and (d) 400X.

- Adenosis
- Fibroadenoma
- Phyllodes Tumor
- Tubular Adenoma

The malignant class contains the following tumor types:

- Carcinoma
- Lobular Carcinoma
- Mucionous Carcinoma
- Papillary Carcinoma

A main characteristic in this dataset is the patient separation. This is something not usually by the patient privacy. The BreakHis dataset includes slide id separation, where each id represents a different slide, or at least a extended and different region of a slide. The compleat split used for all the experiments is represented on table 2.4. These slide separations represents a great opportunity to recreate a real world problem. Due different factors like previously mentioned, patient ethnic origin, sample extraction, hematoxylin and eosin coloration, digitization quality and magnification factor, makes almost impossible to develop a universal model for all the pathological slides. In this case there are real opportunities presented like coloration

Table 2.2: BreakHis Dataset distribution

Magnification	Benign	Malignant	Total
40X	652	1,370	1,995
100X	644	1,437	2,081
200X	623	1,390	2,013
400X	588	1,232	1,820
Total of Images	2,480	5,429	7,909

Table 2.3: BreakHis Dataset distribution for experimentation.

	Training		Test	
Magnification	**Benign**	**Malignant**	**Benign**	**Malignant**
40X	1200	1200	255	490
100X	1200	1200	261	499
200X	1200	1200	255	489
100X	1200	1200	237	418

variation, magnification diversity, patient splitting. There are other considerations that are not a real benefit for model development like the quality of images, the magnifications of 40X and 100X could cover eve a whole slide or important part of one, but the size and resolution are not the best, 700 x 460 pixels when there are some slides of more than 100,000 x 100,000 pixels.

Once presented the advantages and disadvantages of the dataset, two main process where implemented before the experimentation. The first was the data splitting. We verify that between training and testing there were not patient overlapping to replicate a possible real world problem. The patient split were done considering the presence of all the different cancer subtypes on training and testing and always considering maintain a 2 to 1 proportion between training - testing patients and cancer subtypes. These give us an almost 60% - 40% at training - testing split in number of total images. Other common issue that is present is the disproportion between classes. The malignant classes over all the magnification factors is bigger than 2 to 1 over the benign class. To fix this, we repeat a similar technique of LC250000 dataset. Using Augmentor the number of train images between classes was equalized applying left and right rotations up to 12 degrees and a probability of 1. Horizontal and vertical flips were applied with 0.5 probability. Zoom was applied with a max factor of 1.15 with a probability of 0.5 [9]. The final proportion is represented on table 2.3.

The main characteristic in this dataset is the patient separation. This is something not usually due the patient privacy. The BreakHis dataset includes slide id separation, where each id represents a different slide, or at least an extended and different region of a slide. The complete split used for all the experiments is represented on table 2.4. These slide separations represent a great opportunity to recreate a real world problem. Due different factors like previously mentioned, patient ethnic origin, sample extraction, hematoxylin and eosin coloration, digitization quality and magnification factor, makes almost impossible to develop a universal

Table 2.4: Breakhis Dataset patient splitting for experiments.

Training		Testing	
Benign	**Malignant**	**Benign**	**Malignant**
SOB_B_A-14-22549AB	SOB_M_DC-14-10926	SOB_B_A-14-22549G	SOB_M_DC-14-11520
SOB_B_A-14-22549CD	SOB_M_DC-14-11031	SOB_B_A-14-29960CD	SOB_M_DC-14-11951
SOB_B_F-14-14134	SOB_M_DC-14-12312	SOB_B_F-14-21998CD	SOB_M_DC-14-16716
SOB_B_F-14-14134E	SOB_M_DC-14-13412	SOB_B_F-14-23222AB	SOB_M_DC-14-17614
SOB_B_F-14-21998EF	SOB_M_DC-14-13993	SOB_B_F-14-9133	SOB_M_DC-14-17915
SOB_B_F-14-23060AB	SOB_M_DC-14-14015	SOB_B_PT-14-21998AB	SOB_M_DC-14-18650
SOB_B_F-14-23060CD	SOB_M_DC-14-14926	SOB_B_TA-14-15275	SOB_M_DC-14-20636
SOB_B_F-14-25197	SOB_M_DC-14-14946	SOB_B_TA-14-16184	SOB_M_DC-14-2523
SOB_B_F-14-29960AB	SOB_M_DC-14-15572	SOB_B_TA-14-3411F	SOB_M_DC-14-2980
SOB_B_PT-14-22704	SOB_M_DC-14-15696		SOB_M_DC-14-3909
SOB_B_PT-14-29315EF	SOB_M_DC-14-15792		SOB_M_DC-14-4364
SOB_B_TA-14-13200	SOB_M_DC-14-16188		SOB_M_DC-14-6241
SOB_B_TA-14-16184CD	SOB_M_DC-14-16336		SOB_M_LC-14-13412
SOB_B_TA-14-19854C	SOB_M_DC-14-16448		SOB_M_LC-14-15570
SOB_B_TA-14-21978AB	SOB_M_DC-14-16601		SOB_M_MC-14-13413
	SOB_M_DC-14-16875		SOB_M_MC-14-18842D
	SOB_M_DC-14-17901		SOB_M_MC-14-19979C
	SOB_M_DC-14-20629		SOB_M_PC-14-15687B
	SOB_M_DC-14-2773		SOB_M_PC-14-15704
	SOB_M_DC-14-2985		
	SOB_M_DC-14-4372		
	SOB_M_DC-14-5287		
	SOB_M_DC-14-5694		
	SOB_M_DC-14-5695		
	SOB_M_DC-14-8168		
	SOB_M_DC-14-9461		
	SOB_M_LC-14-12204		
	SOB_M_LC-14-15570C		
	SOB_M_LC-14-16196		
	SOB_M_MC-14-10147		
	SOB_M_MC-14-12773		
	SOB_M_MC-14-13418DE		
	SOB_M_MC-14-16456		
	SOB_M_MC-14-18842		
	SOB_M_MC-14-19979		
	SOB_M_PC-14-12465		
	SOB_M_PC-14-19440		
	SOB_M_PC-14-9146		
	SOB_M_PC-15-190EF		

model for all the pathological slides. In this case there are real opportunities presented like coloration variation, magnification diversity, patient splitting. There are other considerations that are not a real benefit for model development like the quality of images, the magnifications of 40X and 100X could cover eve a whole slide or important part of one, but the size and resolution are not the best, 700 x 460 pixels when there are some slides of more than 100,000 x 100,000 pixels.

Once presented the advantages and disadvantages of the dataset, two main process where implemented before the experimentation. The first process was the data splitting. We verify that between training and testing there were not patient overlapping to replicate a possible real-world problem. The patient split was done considering the presence of all the different cancer subtypes on training and testing and always considering maintain a 2 to 1 proportion between training - testing patients and cancer subtypes. That split give us an almost 60% - 40% at training - testing split in number of total images. Other common issue that is present is the disproportion between classes. The malignant classes over all the magnification factors is bigger than 2 to 1 over the benign class. To fix this, we repeat a similar technique of LC250000 dataset. Using Augmentor the number of train images between classes was fixed in similar proportions applying left and right rotations up to 12 degrees and a probability of 1. Horizontal and vertical flips were applied with 0.5 probability. Zoom was applied with a max factor of 1.15 with a probability of 0.5 [9]. The final proportion is represented on table 2.3.

Chapter 3

Deep Learning

Deep Learning is probably the most used branch for image and video problems. Not only for classification, different task can be performed like segmentation, estimation, spoofing, image generation and many more [11][12]. For these problems is common to use convolutional neural networks. These networks are the evolution of conventional neural networks focused specifically to obtain all the possible information of high pixel dimension images. Due the high potential of convolutional neural networks, they are the main option to achieve our problem of histopathological image classification.

3.1 Convolutional Neural Networks

First approach for image classification using convolutional neural networks was presented in 2012 with AlexNet [13]. Since that project, year by year, different architectures for convolution networks have been developed. These architectures have been designed to solve complex problems like ImageNet. Applying techniques to achieve deep knowledge without vanishing weights have developed networks like ResNet or Inception[14]. Even when sounds easy to create a convolution architecture, just applying convolution filters, is not to easy, since vanishing, learning rate and loosing weights are issues that have to been considered in a proportion of the number of filters. It is true that year by year the released networks are deeper and deeper, generating higher hardware requirements to be performed.

A convolutional neural network is an alternative of traditional neural network for image task due the possible number of pixels. It means if we have a 256 x 256 pixels image, which is a very low resolution image, we need to work with a network of 65,536 initial nodes or entry, something complicated if we consider all the calculation per layer, using an low resolution image with poor information. To solve this, convolution layers perform less calculations with more information. A convolution can be explained like a matrix multiplication element by element by a window - filter of specific size. The process is explained at figure 3.1. The filters are the main component of the convolution network, these are like a 'node' of the traditional networks. These filters according to their values or weights, are tuned to detect borders, forms, horizontal or vertical lines and more. Filters are staged at long all the architecture alternately with other filters like max filters o mean filters to reduce the information each possible filtered

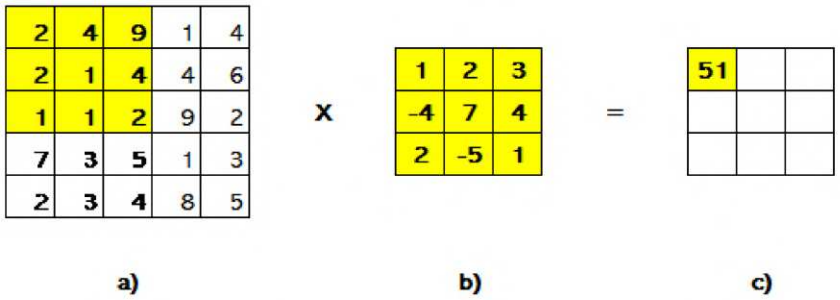

Figure 3.1: Convolution process a) Original image pixel matrix, b) Convolution filter, c) Resultant image matrix filtered.

image. It means that these filters do not multiply weights, they just obtain the max value or the mean value of the window. Once finished the layers training, the final filtered matrix is just transposed to a flatten layer and finish the process like a traditional neural network. There are other modules for deep networks like ResNet named residual module to skip connections, or shortcuts to jump over some layers and prevent vanishing and losing information after a considerable number of layers.

Four our first approximation we decide to use ResNet architecture as baseline. The decision was made considering the problem complexity and the adaptable network depth. ResNet can be adapted to use 18, 34, 50, 101 and 152 layers according to the task complexity. Other main reason to use this architecture is that is well known to achieve good results over digital pathology and image patching[15].

3.1.1 ResNet

As we have mentioned, the main characteristics of ResNet architecture are the skip connections. Each layer can be indexed as $(l-2)$ to (l), or (l) to $(l+2)$, according to the back propagation or forward propagation. Then, the weight matrix of each used filter is represented as $(W^{l-1,l})$, for the weights at layer $(l-1)$ to (l). Forward propagation through the activation function can be defined as the functions 3.1 and 3.2.

$$a^l := g(W^{l-1,l} \cdot a^{l-1} + b^l + W^{l-2,l}) \tag{3.1}$$

$$a^l := g(Z^l + W^{l-2,l} \cdot a^{l-2}) \tag{3.2}$$

Where (a^l), is the output of layer (l) and (g) is the activation function of layer (l). Backward propagation is defined in function 3.3.

$$\Delta w^{l-2,l} := -\eta \frac{\partial E^l}{\partial w^{l-2,l}} = -\eta a^{l-2,l} \cdot \delta^l \tag{3.3}$$

Table 3.1: Patching distribution for datasets.

	LC250000				BreakHis							
	Colon		Lung		40X		100X		200X		400X	
	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
Number of patches	120,545	~130,000	159,568	~30,000	18,000	~14,920	18,000	~15,981	18,000	~16,390	18,000	~13,776

3.1.2 Patching

Classification of hole slide images requires an extended prepossessing. The original and more common format of these slides is svs, from Aperio; a digital hardware and software for digitization. This specific format and high pixel dimensions require a format transformation using openslide library to convert it in a common format like jpg or png. Format process can involve image resize to improve data manageability, but image reduction also means less information. Is recommended use image dimensions according to the hardware possibilities, like memory and processing power. Then, to achieve the hardware limitation is a very common technique apply patching to reduce image dimension without loss information.

Even when we are not working with whole slide images, patching is used to create robust models. Independently of image dimensions, is recommended when data is not enough after data augmentation, and not recommended when image contains loss image concentration or considerable background, for example, persons or animals in landscapes. Since BreakHist dataset contains these conditions, we decide to implement this technique to this and LC25000 dataset. According to the number of classes and data characteristics different configurations were performed for each case.

The selected resolution for patching was 224 x 224 pixels dimensions, since is a very common selection for patching. This resolution is very frequently used for being used in original pretrained models like VGG. Overlapping is also considered to generate more images and increase the robustness of the model. For colon images were used and overlapping with factor of 3 for training and testing, 1 for training and 2 for testing at lung images and finally for BreakHis dataset were used 2 for training and 3 for testing. The patches were filtered applying white space color of 85% at the patch image. The final patch distribution is described at 3.1, patches at testing label are an approximation of the direct division at 224 pixel path sizes multiplied by the overlapping factor, could be less by the white space filter. A sub folder indexation by image/patches was required for final test evaluation, doing hard to calculate the precise number of patches.

3.1.3 Related Work

Convolution networks as baseline is the most common strategy not only for these datasets, probably for all the image classification problems. Patching strategy is more common on whole slide problems. Authors of BreakHis dataset use AlexNet as architecture after having performed some experiments with LeNet architecture[10]. The authors use the different patch sizes 64 x 64 and 32 x 32 pixels using 50% of overlapping factor. The results obtained by the authors are 85.6% accuracy for 40X magnification, 83.5% for 100X, 83.1% and 80.8% for

400X magnification. The dataset was released 4 years ago, where architectures like ResNet or Inception have been published or started to being used at complex problems.

Benhammou et al. performs a model using deep convolution networks, applying Inception V3 [16][17]. In this work, the authors compare different approximations to compare, based on transfer learning and pre-processing impact. Obtaining good results when data augmentation is applied and a considerable worse performance when is not applied or the model is not fully trained in transfer learning. The results obtained in accuracy were 90.2% for 40X magnification, 85.6% at 100X magnification, 86.1 for 200X and 82.5% for 400X magnification.

Finally, Han et al. propose an original model named CSDCNN (Class structure-based deep convolutional neural network) based on GoogLeNet[18][19]. Authors propose a model with multiple hidden layers. Data pre-processing includes data augmentation and image resize to 256 x 256 pixels. The kernel layers are basically compounded by 3 x 3, 5 x 5 and 7 x 7 filers. Convolution kernels are applied in overlapping windows and Gaussian distribution with deviation of 0.01. The last layer is composed by 64 filters obtained through ReLU activation functions. The results at accuracy are 95.8% for 40X magnification, 96.9 at 100X magnification, 96.7 at 200X magnification and 94.9 for 400x magnification.

LC25000 dataset was released at the beginning of this year, there is not related works published yet. There are similar works with lung and colon cancer histopathological images. Wei et al. apply the same strategy using whole slide images of lung cancer[20]. They used ResNet model to work with 476 whole slide images, obtaining a kappa score of 0.525 and an agreement of 66.6% with three pathologists for the classification of predominant patterns. The patterns are lepidic, acinar, papillary. micropapillary, solid and benign. For the specific patch classification of the 6 different class were obtained AUC's bigger than 98%. This work is a great demonstration of why image patching is a great strategy when it is necessary to work with high dimensions and quality images.

Basha et al. performs a colon cancer nuclei classification using convolutional neural networks[21]. The authors propose and efficient model names RCCNet. The model is compound by more than 1.5 million learnable parameters. Authors obtain an accuracy of 80.61% and a .7887 weighted average F1 score. Accuracy and F1 scores obtained are better than other complex network architectures like GoogLeNet and AlexNet. The author mention that the network is highly efficient due the use of less parameter and better outcomes. Some of the main keys for model performance are a input 32 x 32 pixels and a continue cycle of convolutions, pulling layers and fully connected layers, the main novelty of the work is that practically, the input of the following blocks at the models are the flatten features of the last pulling layer.

3.1.4 Results

Results are obtained using different architectures of ResNet according to the complexity of the problem. For example, colon dataset was our first experiment, to obtain concrete results

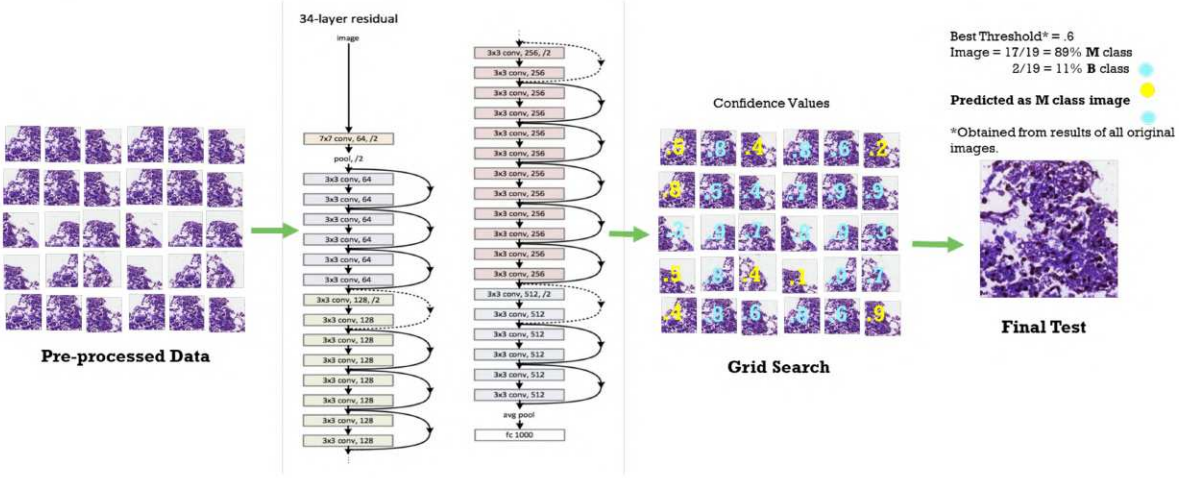

Figure 3.2: Convolutional Neural Networks strategy.

ResNet50 and overlapping of 3 were implemented. Results were perfect, but considering the problem, probably, this implementation was over-saturated. We need to remember that LC25000 dataset is less complex than BreakHis because each class contains a specific tissue type and not 4 different tissue types by class like BreakHis. All the images come from the same distribution, they have a good resolution and there is not a specific patient separation on them. Then, for these reasons, in lung implementation ResNet34 was selected using a smaller overlapping. For the same reasons mentioned previously, ResNet152 was selected for BreakHis dataset. For all the implementations each model was trained with 50 iterations, Adam optimizer with the default parameters, learning rate of 0.001, $\beta 1 = 0.9$, $\beta 2 = 0.999$ and using a softmax output functions from equations 3.4 to 3.8. For this and the rest of the tables keywords are aca for adenocarcinoma, n for benign and scc for squamous.

$$m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t \tag{3.4}$$

$$v_t = \beta_2 \cdot m_{t-1} + (1 - \beta_2) \cdot g_t^2 \tag{3.5}$$

$$\hat{m}_t = \frac{m_t}{(1 - \beta_1^t)} \tag{3.6}$$

$$\hat{v}_t = \frac{v_t}{(1 - \beta_2^t)} \tag{3.7}$$

$$W_t = W_{t-1} - \eta \cdot \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{3.8}$$

Each model was trained with all the training patches. The test patches were separated by folder, each patch is predicted by the model, obtaining a value between 0 and 1 for each class. Then main advantage of a robust model is the possibility of applying a threshold if the

Table 3.2: Accuracy and area under the curve for convolutional neural network experiments.

Dataset	Accuracy	AUC
Colon	1	1
Lung	.948	.98
Breast 40X	.887	1
Breast 100X	.85	1
Breast 200X	.887	1
Breast 400X	.835	1
Average	***.901***	***.996***

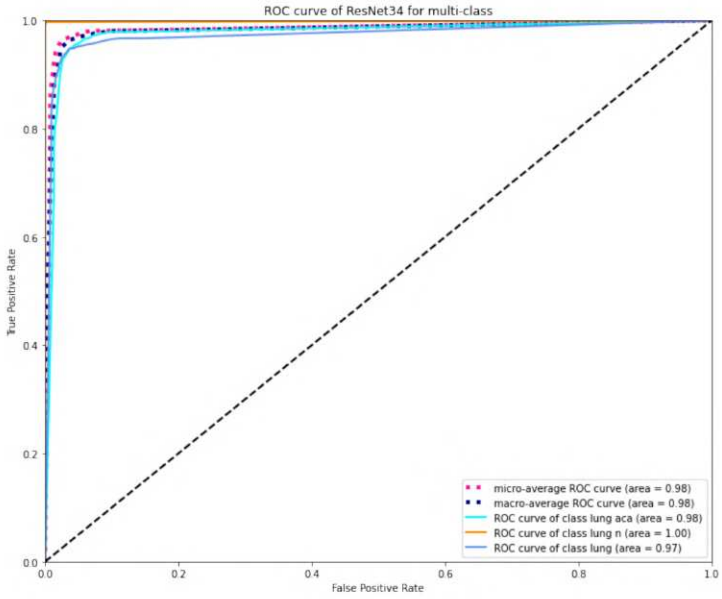

Figure 3.3: Lung ROC curve.

final prediction is done by voting. It means, an original image is divided by smaller patches, is patch has his own predicted value, according to the threshold, if the predicted value is higher, it votes. In nutshells, the main idea is that the patches with the highest predictions can defined the final class of the image. Nine thresholds are used [.1, .2, ..., .8, .9], each threshold is evaluated and then the best outcome is selected. This process is explained on the figure 3.2.

We can make partial conclusions results on table 3.3 and 3.3. For example, that colon images can be perfectly separable from benignant to adenocarcinoma. For lung images, results are convincing, but the ROC curve says that maybe also could be perfectly separable. In the case of BreakHis dataset we obtain better results than the original author, and closer to the Benhammou et al. [16]. But the outcomes are far from the obtained by Han et. al. [18]. The magnification factor 200x looks like the best option for the problem, and the rest of magnifications obtain except the 100X magnification obtain similar results. In specific for lung images the AUC was selected by the micro average curve of the 3 classes. For BreakHis dataset the ROC curve could be considered like a better option to measure the model performance due the

high disproportion of images at test set. Unfortunately these outcomes cannot be considered seriously because there were obtained previously from a patch threshold, obtained almost binary classifications of 0 and 1. It will be necessary another strategy to improve the accuracy and obtain clear conclusions of the ROC curve.

3.2 Transfer Learning

Transfer learning is a very helpfully technique when we have trained a model for a similar problem. For example, if we have trained a model to classify dogs and cats, it could be use full for classification of wolfs and wildcats. Since the model has been trained to learn different forms like edges or curvatures present on these animals, maybe could not be necessary train a new model. Some of the most common techniques involve weights trained with ImageNet problem. Since we have mentioned before, ImageNet is a complex problem consisting in the classification of 1,000 classes like persons, animals, food, plants, rooms, cars and different tools and instruments. Then, these weights trained for identify these classes can help for specific problems, maybe belonging to some of the 1,000 classes. Then, is a very common technique initialize the model weights with that trained weights. Almost all the popular architectures, or at least all the architectures that have performed the state of the art on this kind of problems have an ImageNet weigh initialization option.

3.2.1 Related Work

Bayramoglu et al. perform transfer learning experiments for cell nuclei classification in histopathology images [22]. The author tackles the data limitation with 4 different architectures, AlexNet, GoogLeNet, GenderNet and VGG-16. For experimentation, the authors use HistoPhenotypes dataset, which contains images from 4 different classes, epithelial, inflammatory, fibroblast and miscellaneous. Last category consists of mixed cell nuclei, then this is excluded from the training. All the architectures were trained with the same parameters and same iterations, with two variations, once initialized with random weights, and second with ImageNet weights. For each iteration, the performance in validation accuracy is presented. A big difference is performed in AlexNet, where random weighted model after 200 iterations achieve same results than weighted models. Like the rest architectures with less difference [23]. The author concludes that transfer learning helps in a considerable way for training performance.

Other important related work with transfer learning for digital pathology is presented by Shin et al. for classification of thoraco-abdominal lymph nodes and interstitial lung [24]. Study is developed over CifarNet, AlexNet and GoogLeNet with different model training parameter values. CifarNet is used as baseline for lymph nodes the rest of networks are modified to evaluate state of the art at ImageNet problem. Outcomes where very conclusive obtaining similar accuracy performances ten times faster when ImageNet weights are utilized. Authors conclude that transfer learning achieves the best performance. Adding that optimal learning rate for many layers is a big challenge, especially for deep networks like GoogLeNet.

3.2.2 Results

Since we have used a baseline in the previous implementation, in this experiment we will apply direct transfer learning. It means that features will be obtained directly from ImageNet training. In the next experiments we will apply fine tuning to achieve a complete review of convolutional neural networks for digital pathology problem.

VGG19

Simonyan and Zisserman present the VGG architecture in 2015, an architecture for large scale visual recognition [25]. The model is presented at two different depth levels, first VGG16 and second VGG19, the names are assigned by the number of layers, we can see an example at figure 3.4. Even when the network is present as a deep network, for the number of parameters and layer nowadays is considered like some of the first option to test a problem before trying with deeper networks like Inception or ResNet. To reduce the number of parameters, the authors use 3 x 3 filters in all the convolutions. For our experiments we use VGG19. The model is initialized with ImageNet weights and after one iteration, features obtained by the last flatten layer are trained using different machine learning techniques.

- K - nearest neighbors
- Stochastic gradient decent
- Naive Bayes
- Decision tree
- Adaboost
- Gradient boosting
- Random forest
- Extremely trees
- Linear support vector machine
- Support vector machine

K - nearest neighbors is trained with 3 different K's, 3, 6 and 9, and the best model with higher accuracy is selected. **Stochastic gradient decent** used hinge loss function and penalty of 12. **Naive Bayes** model used variable smoothing of $1e^{-9}$. For **Decision tree**, Gini function was selected as measure for quality split. **Adaboost** used 100 as maximum estimators, learning rate of 1 and SAMME.R as boosting algorithm. **Gradient boosting** used 200

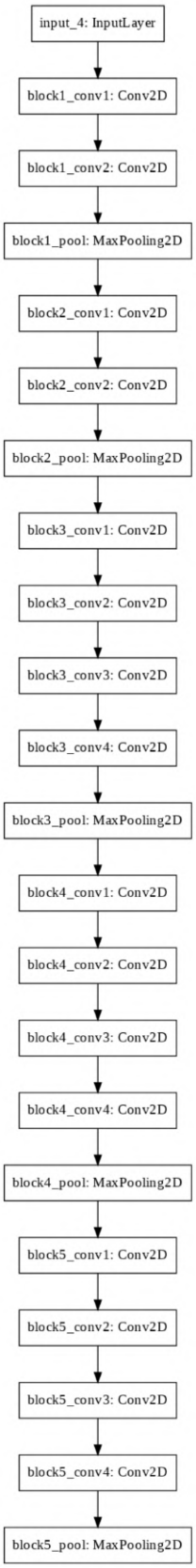

Figure 3.4: VGG19 model.

Table 3.3: Accuracy and area under the curve for LC25000 dataset using VGG19 transfer learning.

Technique	Accuracy		AUC	
	Lung	Colon	Lung	Colon
KNN	0.983	**0.995**	1	**1**
SGD	0.917	0.991	0.99	1
Naive Bayes	0.853	0.95	0.89	0.98
Decision Tree	0.868	0.926	0.89	0.93
Adaboost	0.872	0.988	0.99	1
Gradient Boosting	0.937	0.986	0.99	1
Random Forest	0.946	0.988	0.99	1
Extremely Trees	0.952	0.988	0.99	1
Linear SVM	0.949	0.990	0.99	1
SVM	**0.983**	0.975	**1**	1
Average	***0.93***	***.977***	***.972***	***.991***

as maximum estimators, random state of 0, learning rate and max depth of 1. **Random forest** and **Extremelly random forest** were trained using maximum 200 estimators, Gini function and minimum split of 2. **Support vector machine** and **Linear support vector machine** were trained using different C values, [0.01, 0.1, 1, 10, 100, 500, 1000, 2000, 4000], in some complex cases like features extracted from very deep networks like Inception, maximum C value was 100. The model with best accuracy was selected for the reported outcomes.

We can see for the results on table 3.3 and figure 3.6 that there is a great performance for LC2500 dataset using transfer learning with machine learning techniques. We have mentioned previously that this has sense since the color distribution, resolution, and specific tissue classes. Deep learning and machine learning are probably better option than just deep learning. Transfer learning looks like a very viable option to test a problem. If we compare this implementation with our previous implementation, this is faster, easier and cheaper talking about hardware requirements. Even, results are better for lung images, achieving an accuracy of 98.36%, 4% better than robust convolution neural networks. This is just the case of LC25000 dataset. For BreakHis dataset, results on table 3.4 and figure 3.7 do not show an improvement from the robust model implementation. It could sound logic due the issued that we have mentioned before. But that does not mean that there is not any possibility to at least achieve the convolution neural network as baseline with a less complex option.

To have a better appreciation, a T-SNE representation was done for all the features extracted on this project[26]. These representations were done using training features extracted from different convolution architectures. T-SNE configuration includes 2 components, learning rate = 150, perplexity = 30 and an angle of 0.2. This algorithm and machine learning techniques were implemented using scikit-learn [27]. For example at figure 3.5, we can see a great feature representation of LC25000 dataset and why its classes are almost perfectly separable.

Table 3.4: Accuracy and area under the curve for BreakHis dataset using VGG19 transfer learning.

Technique	Accuracy				AUC			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
KNN	0.774	0.734	0.754	0.737	0.79	0.74	0.79	0.78
SGD	0.783	0.792	0.783	0.772	0.78	0.81	0.84	0.8
Naive Bayes	0.715	0.693	0.783	0.68	0.77	0.75	0.84	0.8
Decision Tree	0.715	0.698	0.661	0.691	0.7	0.67	0.65	0.69
Adaboost	0.775	0.768	0.791	0.772	0.82	0.84	0.86	0.84
Gradient Boosting	0.766	0.777	0.803	0.772	0.82	0.85	0.87	0.84
Random Forest	0.798	0.786	0.852	0.812	0.86	0.87	0.91	0.89
Extremely Trees	0.808	0.784	**0.86**	0.81	0.87	0.87	0.91	**0.89**
Linear SVM	0.793	**0.817**	0.817	0.77	0.86	0.86	0.9	0.85
SVM	**0.833**	0.793	0.842	**0.81**	**0.89**	**0.87**	**0.92**	0.88
Average	***0.776***	***0.7642***	***0.7946***	***0.7626***	***0.816***	***0.813***	***0.849***	***0.826***

The benefit of transfer learning and his high impact over complex problems is demonstrated on this example. Even when results on breast images are not good, to being results obtained from just one iteration they are very impressive. We can observe that machine learning techniques have similar performance when they are implemented over high quality features like colon or lung features. In other cases, like breast feature the difference is higher between support vector machines, extremely trees and the rest of techniques. Finally, since VGG19 outcomes for BreakHis dataset were not satisfactory, deeper networks will be tested for these images.

ResNet152

Trying to achieve better performance from the technique in BreakHist dataset we decide to implement deeper networks. ResNet152 was used to compare the performance of the network using the model as baseline or transfer learning. Similar to VGG19, all the model were initialized with ImageNet weights, and features were extracted using the flatten layer and used to train 10 machine learning techniques with the same parameters used for VGG19 and all the project.

For the outcomes on table 3.5 we can establish direct transfer learning using very deep networks could not be a viable option. First, we need to remember that an inconvenient of these architectures is the vanishing weight rate increase with each layer. Then, is tried to solve the issue with the skipped weight from previous layer to deeper layers. This can be helpfully after some iterations to keep the impact factor of previous weights, but then what happen when is there only one iteration to obtain the features. Probably the vanishing will still at the end of flatten layer. Then features extracted will be a worse version of features that maybe could be extracted at the half of the model layers.

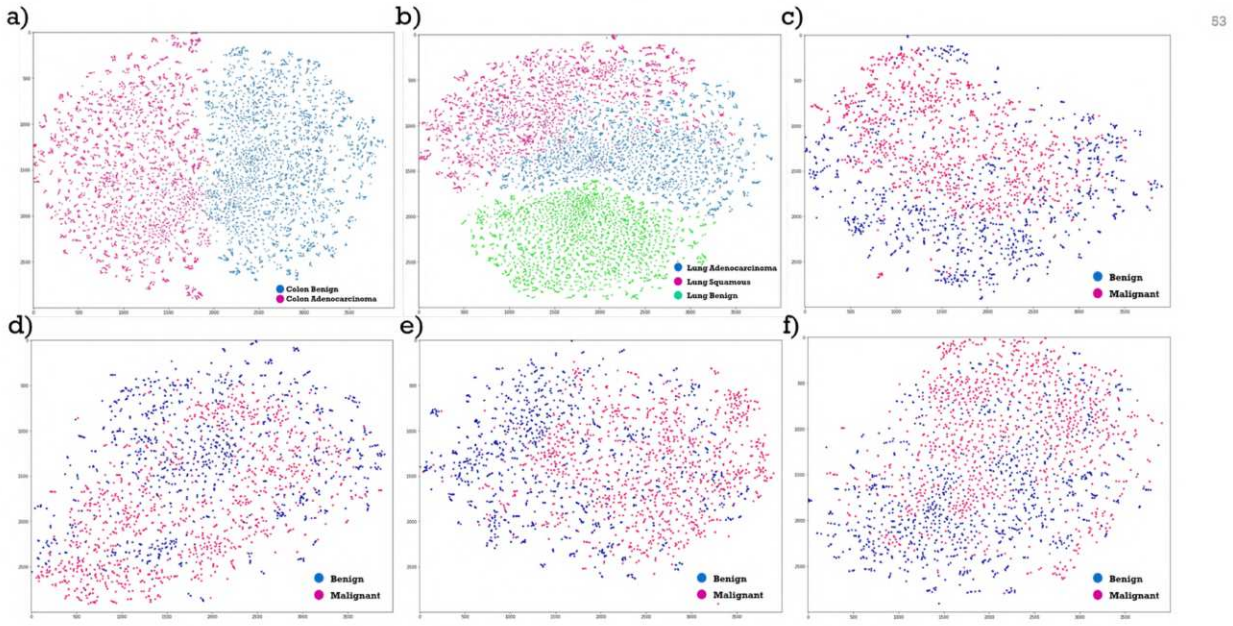

Figure 3.5: VGG19 T-SNE representation. a) Colon b) Lung c) 40X Breast d) 100X Breast e) 200X Breast f) 400X Breast.

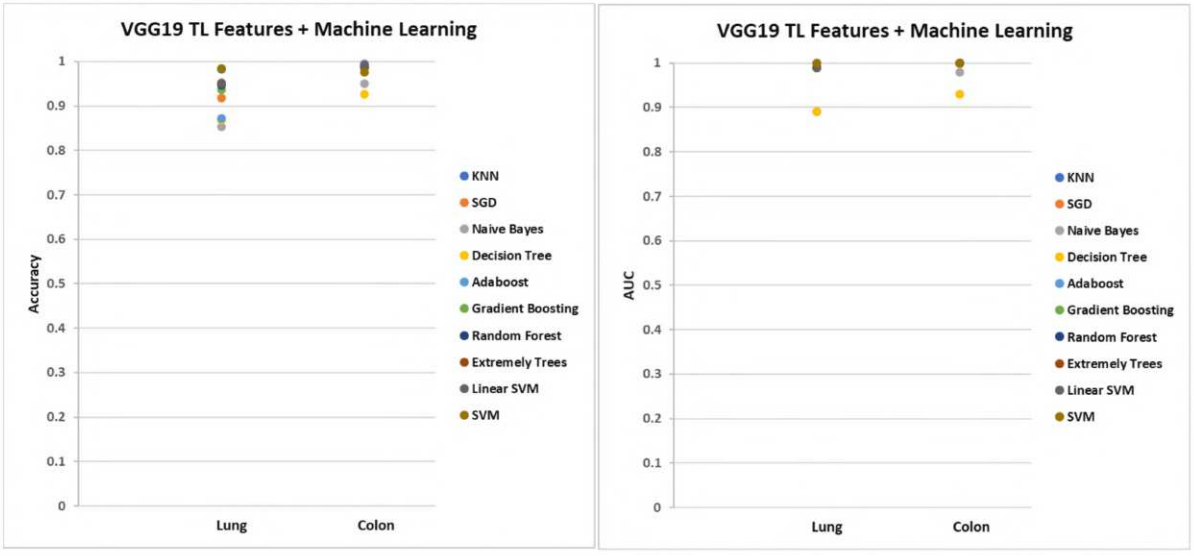

Figure 3.6: LC25000 machine learning performance using VGG19 features with transfer learning.

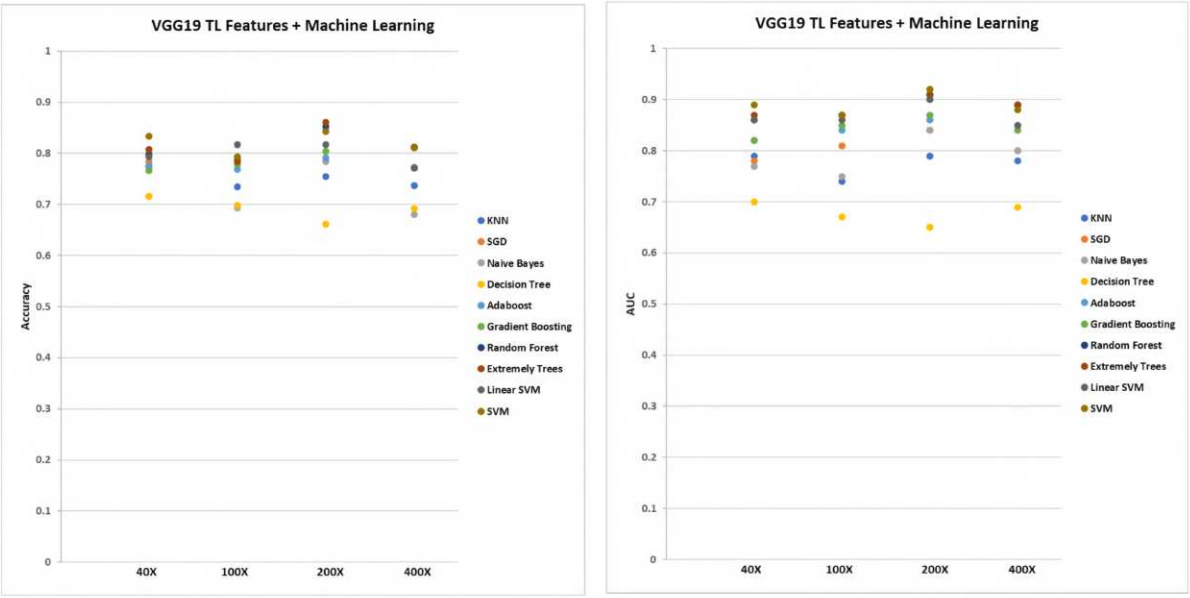

Figure 3.7: BreakHis machine learning performance using VGG19 features with transfer learning.

Performance of the strategy using ResNet152 model for transfer learning does not look like the best option. If we compare VGG19 and ResNet152 as feature extractor, VG119 is better not only for the accuracy and AUC achieved, also because it has considerably less parameters than ResNet152. This is not a big disadvantage or error, since the inversion of time or resources is not of a baseline. That could take more time to obtain the features, but even when there was many present layers at ResNet152, no one of them were trained. That could give us a clue that is more practical not to use very deep networks to extract features.

Inception V3

Finally, the last model to experiment is Inception V3[28]. This architecture has been very popular since his publication in 2016. Like ResNet152, this architecture is a very deep network, but with a great improvement in parameters. The first version of this model has 23 million of parameters around 159 layers. If we make a comparison, VGG19 has 143 million and 26 layers and ResNet152 has 60 million. Inception V3 network has 23 million of parameters over 159 layers. That has been the most exploded advantage of this model and the main reason to select it for complex problems. It is a combination of deep layers with less parameters. The architecture began like module modification of GoogLeNet, the main characteristics were 3 x 3 convolutions instead of 5 x 5, 3 x 1 convolution after 1 x 3 convolution instead of 3 x 3 convolution. Module B of the network changes 7 x 7 filters by 2 of 1 x 7 and 7 x 1. One auxiliary classifier instead of two was implemented using 1 x 1, 5 x 5 and 5 x 5 convolutions. Simple but very influential changes make possible a deeper network with high reduction of parameters.

Table 3.5: Accuracy and area under the curve for BreakHis dataset using ResNet152 transfer learning.

Technique	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.692	0.682	0.701	0.654	0.71	0.71	0.74	0.68
SGD	0.659	0.694	0.674	0.61	0.73	0.76	0.8	0.74
Naive Bayes	0.609	0.628	0.665	0.629	0.68	0.67	0.72	0.7
Decision Tree	0.587	0.65	0.677	0.648	0.59	0.63	0.65	0.62
Adaboost	0.681	0.713	0.741	0.69	0.7	0.75	0.8	0.76
Gradient Boosting	0.695	0.709	0.735	0.682	0.71	0.75	0.8	0.75
Random Forest	0.7	0.73	0.748	**0.696**	0.76	0.79	0.83	0.75
Extremely Trees	0.688	0.727	0.745	0.687	0.76	0.78	0.82	0.75
Linear SVM	**0.75**	**0.736**	**0.77**	0.694	**0.8**	**0.79**	**0.83**	**0.77**
SVM	0.657	0.673	0.657	0.668	0.7	0.73	0.75	0.72
Average	***0.6718***	***0.6942***	***0.7113***	***0.6658***	***0.714***	***0.736***	***0.774***	***0.724***

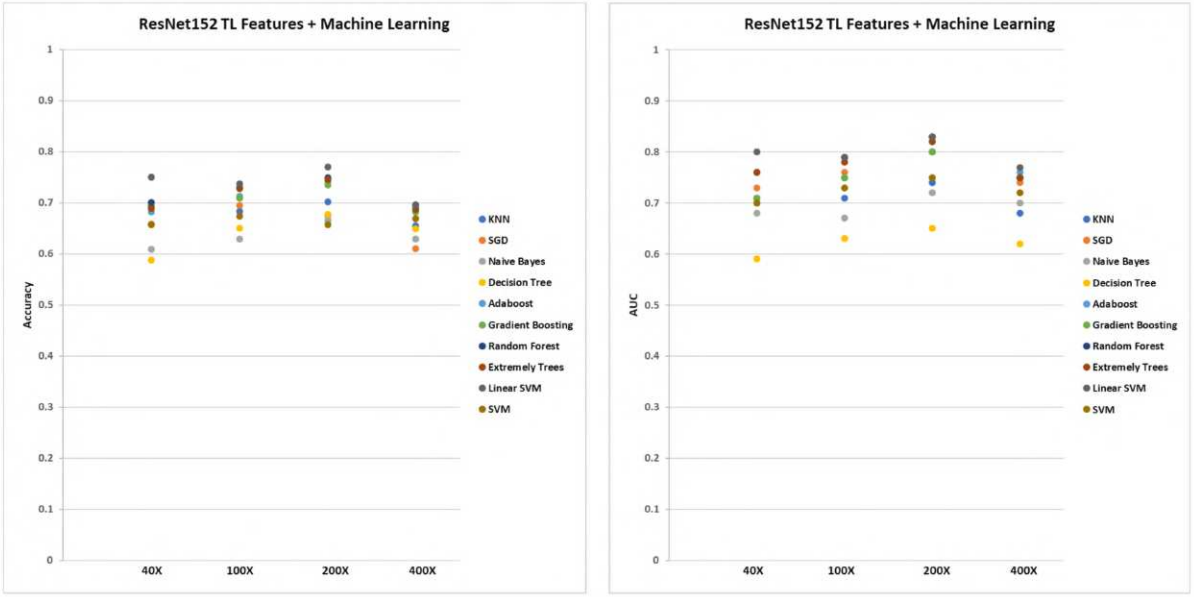

Figure 3.8: BreakHis machine learning performance using ResNet152 features with transfer learning.

Table 3.6: Accuracy and area under the curve for BreakHis dataset using Inception V3 transfer learning.

Technique	Accuracy				AUC			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
KNN	0.504	0.552	0.518	0.497	0.5	0.52	0.52	0.46
SGD	0.512	0.396	0.654	0.532	0.44	0.45	0.55	0.41
Naive Bayes	0.532	0.58	0.537	0.442	0.46	0.46	0.57	0.44
Decision Tree	0.547	0.581	0.526	0.529	0.52	0.53	0.53	0.48
Adaboost	0.496	0.593	0.561	0.483	0.47	0.57	0.57	0.45
Gradient Boosting	0.507	0.589	0.553	0.528	0.46	0.56	0.53	0.5
Random Forest	0.559	0.577	0.533	0.551	0.52	0.56	0.55	0.53
Extremely Trees	0.551	0.571	0.53	0.561	0.53	0.53	0.55	0.54
Linear SVM	0.657	0.656	0.657	**0.638**	0.47	0.47	0.57	**0.56**
SVM	**0.657**	**0.656**	**0.657**	0.638	**0.5**	**0.51**	**0.57**	0.5
Average	***0.5522***	***0.5751***	***0.5726***	***0.5399***	***0.487***	***0.516***	***0.551***	***0.487***

Surprisingly, results obtained by Inception V3 network on table 3.6 were the worst. There could be different possible reasons, starting for the resolution, network is designed for image vision using high resolution images. This is the deepest network, but also is the network with the smallest filters, something that is viable when there is presence of information in big part of the image. In our case due the different magnifications and median resolution filter size experiment does not take any advantage of the filters size. Finally, we can conclude that maybe is possible doing similar experiments before establishing a final conclusion of datasets. For example, fine tuning could be an alternative, needing training this time, but not a complete training, converting again deep learning like a possible alternative for the problem.

Before start with the following experiments, we want to be sure of the performance and scope of transfer learning, at least for this dataset. Then, we decide to implement a combination of features, due the accessibility in resources and time for experiments, we combine the models with better accuracy, which are VGG19 and ResNet. The implementation was quite easy, we only combine the features of the two models and then were training on the machine learning techniques. Similar, we implement a model using 3 models, VGG19, ResNet and Inception V3.

Outcomes on table 3.7 were very promising, for the first combination we obtain better results than the obtained from each model. Accuracy and area under the curve were quite close from the obtained using the patching baseline, in almost all the magnifications at just 1%. In the case of the whole model combination, outcomes of table 3.8 were practically the same, doing a null improvement of Inception V3 model. From the first combination we can say that we are close to find an implementation capably to tackle digital pathology in an efficient way. Fine tuning sounds like that little improvement necessary to achieve the desire performance.

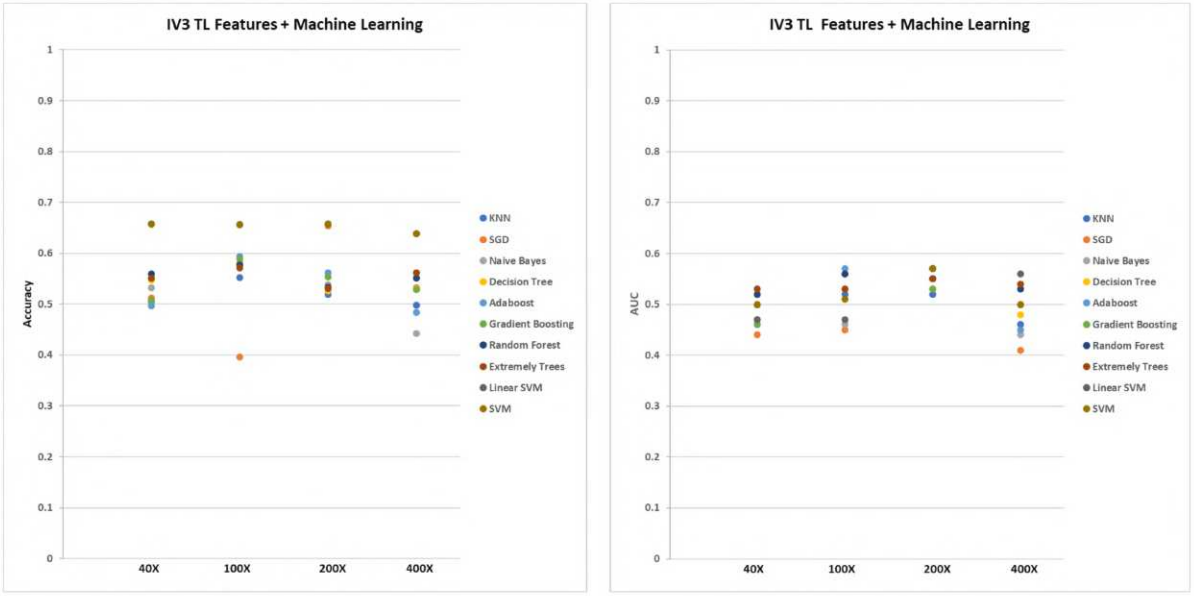

Figure 3.9: BreakHis machine learning performance using Inception V3 features with transfer learning.

Table 3.7: Accuracy and area under the curve for BreakHis dataset using VGG19 + ResNet152 transfer learning.

Technique	Accuracy				AUC			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
KNN	0.771	0.734	0.754	0.737	0.79	0.74	0.79	0.78
SGD	0.781	0.782	0.733	0.716	0.82	0.75	0.85	0.77
Naive Bayes	0.656	0.657	0.772	0.694	0.74	0.71	0.81	0.79
Decision Tree	0.692	0.707	0.728	0.699	0.69	0.64	0.72	0.7
Adaboost	0.762	0.797	0.817	0.792	0.82	0.86	0.88	0.85
Gradient Boosting	0.765	0.784	0.823	0.787	0.82	0.87	0.89	0.85
Random Forest	0.791	0.807	0.841	0.772	0.86	0.88	0.91	0.84
Extremely Trees	0.789	0.803	0.837	0.757	0.86	0.88	0.91	0.84
Linear SVM	0.793	0.81	0.818	0.77	0.86	0.87	0.9	0.85
SVM	**0.845**	**0.813**	**0.864**	**0.83**	**0.9**	**0.89**	**0.93**	**0.89**
Average	***0.7645***	***0.7694***	***0.7987***	***0.7554***	***0.816***	***0.809***	***0.859***	***0.816***

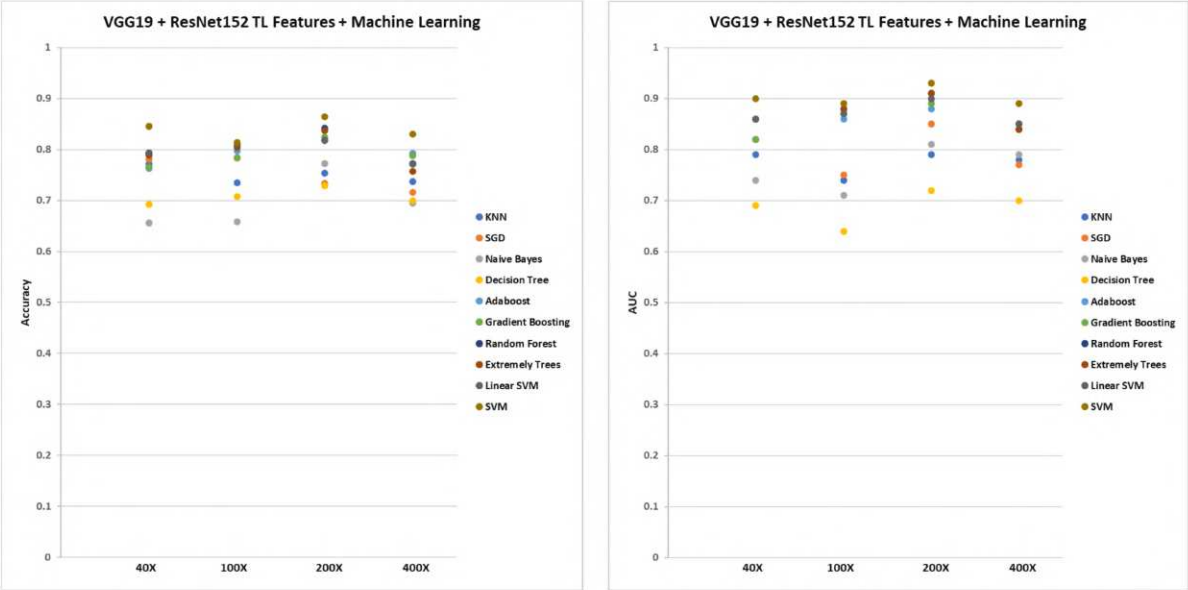

Figure 3.10: BreakHis machine learning performance using VGG19 + ResNet152 features with transfer learning.

Table 3.8: Accuracy and area under the curve for BreakHis dataset using VGG19 + ResNet152 + IV3 transfer learning.

Technique	Accuracy				AUC			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
KNN	0.771	0.734	0.754	0.737	0.79	0.74	0.79	0.78
SGD	0.676	0.817	0.72	0.757	0.8	0.84	0.81	0.79
Naive Bayes	0.645	0.655	0.76	0.667	0.69	0.71	0.8	0.74
Decision Tree	0.695	0.678	0.735	0.69	0.7	0.68	0.73	0.69
Adaboost	0.747	0.809	0.803	0.78	0.81	0.86	0.89	0.84
Gradient Boosting	0.763	0.801	0.836	0.774	0.81	0.87	0.9	0.83
Random Forest	0.769	0.803	0.833	0.764	0.85	0.87	0.91	0.84
Extremely Trees	0.769	0.786	0.829	0.758	0.85	0.87	0.91	0.83
Linear SVM	0.794	0.811	0.818	0.77	0.86	0.87	0.89	0.85
SVM	**0.845**	**0.811**	**0.868**	**0.829**	**0.9**	**0.89**	**0.93**	**0.89**
Average	***0.7474***	***0.7705***	***0.7956***	***0.7526***	***0.806***	***0.82***	***0.856***	***0.808***

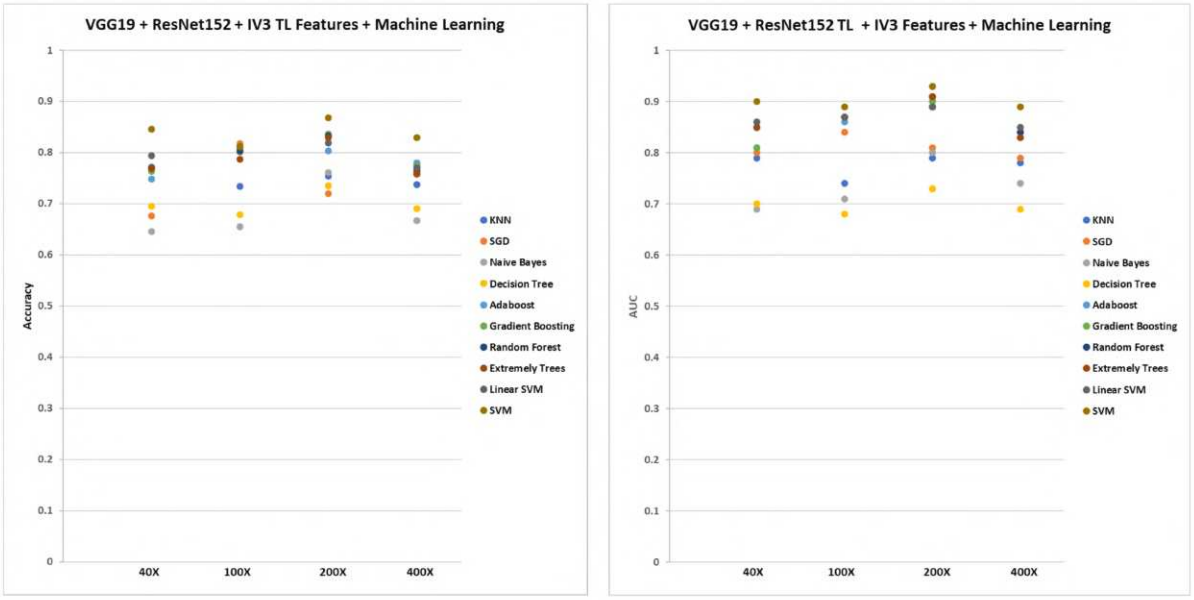

Figure 3.11: BreakHis machine learning performance using VGG19 + ResNet152 + IV3 features with transfer learning.

3.3 Fine Tuning

Fine tuning is a common technique used for efficient deep learning application. since we have mention before, there was different ways to use transfer learning according to the similarity of task. For example, some persons establish transfer learning just like weight initialization, other like use of weights to obtain direct prediction. In our case we decide to use the last option. Then, fine tuning could be seen like a deeper option or lighter option. The technique consists basically in training just a part of the layers in the model. These layers need to be pre-trained with weights of similar problems. The rest of the layers are initialized with the same weights, but they change and fitting iteration by iteration. Then for our purpose, this is a deeper technique than transfer learning, because in this case we need to train the model.

3.3.1 Related Work

Tajbakhsh et al. presents a great comparison between full training and fine tuning performance using medical images [29]. Experiments were realized using images from 3 medical areas, radiology, cardiology and gastroenterology. Performance were measured using task like classification, detection and segmentation. AlexNet is the architecture is sued to being trained from scratch and trained using fine tuning. Models were trained initializing with ImageNet weights. Different combinations were tested, there was a total of 8 tuning combinations and the scratch training. These combinations begin from the 1st convolution block until the last fully connected layer, then from second convolution block to last fully connected layer and progressively until the last combination that only trains the last fully connected layer. Outcomes were very interesting, for almost all the experiments the combinations with better area

under the curve were conv5-fc8, conv1-fc8 and scratch, in that order. Authors conclude that use of a pre-trained models with adequate fine tuning proportion outperformed or, in the worst case, performed as well as model trained from scratch. Fine tuned models were more robust to the size of training than the trained from scratch. Finally, layer wise fine tuning could be a practical way to reach a great performance for the application at hand based on the amount of available data.

Kumar et al. present a work using an assemble of convolutional networks for the ImageCLEF 2016 dataset [30]. Authors uses two architectures AlexNet and GooLeNet, both initialized with ImageNet weights. The strategy applied to the modes is use the feature extracted from the last flatten layer and used in different ways. One of them is softtmax classifier using the same convolutional network. Second use of the features is to train a one vs one multi class support vector machine. And finally, a combination of features extracted from both models to train a support vector machine model. At final comparison, the best model was that which use features extracted from both models and support vector machines, had the best accuracy, followed by the fine tuned with softmax, then fine tuned with support vector machines and at the end transfer learning and support vector machines. In this work we can obtain a probe that model mixture could be a great technique with low requirement cost using transfer learning or fine tuning.

Finally, Qu et al. published a work of gastric pathology classification using stepwise fine tuning [31]. The project is divided in fine step wise tuning stage using tissue-wise data and step wise fine tuning using cell wise data. For each part, three architectures are used, VGG16, AlextNet and Inception V3. For all architectures, the last flatten layer was trained for classification. For training stages, the accuracy increases in 5% after the second state. It means that the model was trained using tissue wise and then a final evaluation was done after retraining the same model with cell wise images. The best model for this implementation was de VGG16 architecture. The author concludes that fine tuning is an important technique that allows the use of deep learning in problems like digital pathology, where images have high dimension and dataset can be very extensive, in this case the tissue data was compound of 45,000 images of 3 different classes of gastric tissue, and cell wise data compound of almost 13,000 images. Training from scratch 3 models, including very deep architectures like Inception v3 could take more time and resources than the same experiments using fine tuning.

3.3.2 Results

From related work we have obtained interesting techniques. For example, it is probed that combination of models with machine learning has been a great solution for similar problems. Also, the selection of networks is quite similar for the selected for our experiments. Finally, we have seen that fine tuning is a great strategy but also it needs considerable experimentation to obtain conclusions. Some of these results consist in testing different trainable layer lapses. In our case, we are looking for a real alternative for deep learning, which means that we cannot consider as an alternative for a full model trained a model which has been trained at 90% or 80%. Then, similar parameters were selected for the architectures, representing different

deep levels.

Other conclusion obtained from related work is that of fine tuning cannot be the best option if there is not a correct balance between the obtained knowledge and the desire knowledge. ImageNet does not have any relation with our data but it is has demonstrated that ImageNet is a great way to initialize the model weights. Fine tuning allows us to use part if that weighs to obtain some partial 'conclusions' of the model than later will be tuned for the specific model modifying that weights. Then, if the proportion of fixed weights and learning weights is the appropriated, performance at the fixed layers of the weights could not be the best, trying to being improved at the non fixed layers. If there are not enough layers to fix the weights, at the end could be just random weights on the output.

Like transfer learning, this strategy will be dedicated for the outcome improvement of BreakHis dataset. For all the architectures the same training parameters will be used. Training optimizer is RMSprop using learning rate of 0.001, e of $1e-7$ and discounting factor of 0.9, using categorical cross-entropy of functions 3.9 to 3.11. All models will be trained using 15 iterations and extracting the features from the same features at transfer learning. Features will be used to train machine learning techniques.

$$V_{dw} = \beta \cdot V_{dw} + (1-\beta) \cdot dw^2 \tag{3.9}$$
$$V_{db} = \beta \cdot V_{dw} + (1-\beta) \cdot db^2 \tag{3.10}$$

$$W = W - \alpha \cdot \frac{dw}{\sqrt{V_{dw}} + \epsilon} \tag{3.11}$$
$$b = b - \alpha \cdot \frac{db}{\sqrt{V_{db}} + \epsilon} \tag{3.12}$$

VGG19

We have mentioned that VGG19 is one of the popular options to idealize the complexity of a problem, is not considered one of the deepest networks even when it has more trainable parameters than the rest of architectures. VGG19 architecture is defined at main 5 blocks, each block consists in 2 or 4 convolutions followed by a max pooling. For example, at block 1 and block 2, there are 2 convolutions followed by one max pooling layer, at block 3, 4 and 5 there are 4 convolutions followed by one max pooling layer.

If we really want to consider fine tuning as a real alternative, we need to take in consideration a moderate trainable part of the architectures. For VGG19 we decide to train just the last block of the model, that means training approximately 20% of the model, something that we consider a good option considering that is the architecture with more trainable parameters.

Results on 3.9, different to improve, were worse. There are in some cases differences of 10% between the accuracy and area under the curve obtained between transfer learning and

Table 3.9: Accuracy and area under the curve for BreakHis dataset using VGG19 fine tuning.

Technique	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.755	0.696	0.696	0.661	0.77	0.73	0.68	0.7
SGD	0.758	0.672	0.745	0.697	0.73	0.77	0.55	0.78
Naive Bayes	0.769	0.668	0.668	0.665	0.71	0.81	0.53	0.6
Decision Tree	0.699	0.693	0.674	0.629	0.7	0.69	0.63	0.65
Adaboost	0.735	0.726	0.725	0.694	0.79	0.79	0.75	0.77
Gradient Boosting	0.742	0.705	0.727	0.702	0.8	0.76	0.74	0.77
Random Forest	0.769	0.747	**0.763**	**0.741**	0.83	0.83	**0.79**	0.78
Extremely Trees	**0.775**	**0.755**	0.759	0.72	**0.83**	**0.84**	0.78	0.77
Linear SVM	0.751	0.7	0.759	0.719	0.79	0.78	0.79	**0.79**
SVM	0.657	0.656	0.657	0.722	0.47	0.68	0.54	0.79
Average	***0.741***	***0.7018***	***0.7173***	***0.695***	***0.742***	***0.768***	***0.678***	***0.74***

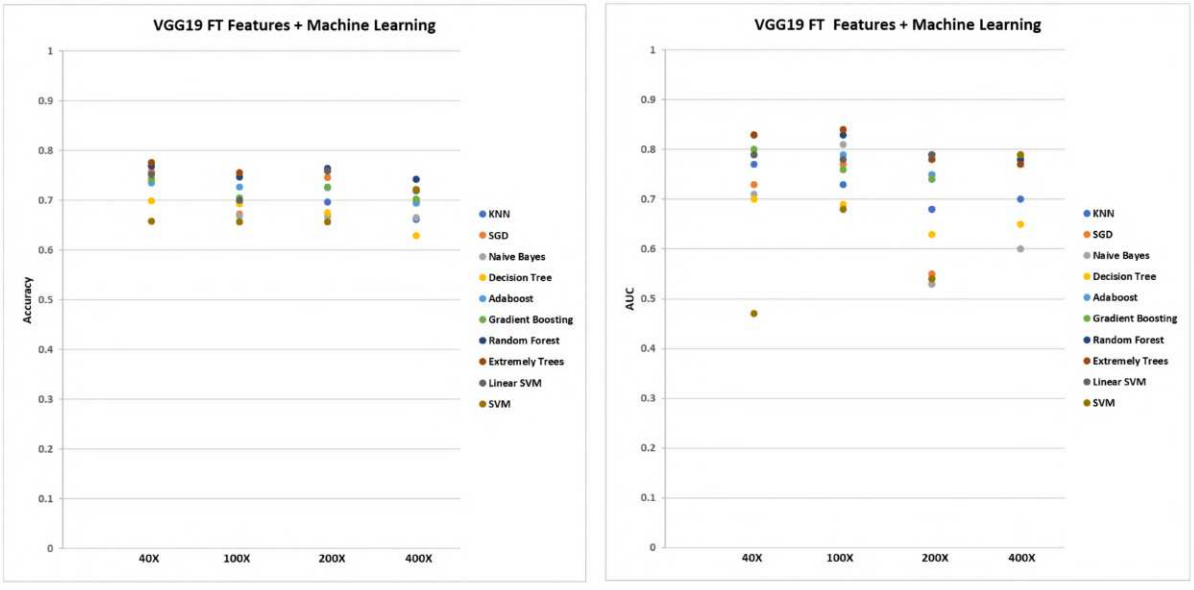

Figure 3.12: BreakHis machine learning performance using VGG19 features with fine tuning.

Table 3.10: Accuracy and area under the curve for BreakHis dataset using ResNet152 fine tuning.

Technique	Accuracy				AUC			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
KNN	0.62	0.635	0.772	0.604	0.63	0.64	0.81	0.61
SGD	**0.668**	0.659	0.634	0.639	0.5	0.7	0.77	0.52
Naive Bayes	0.57	0.617	0.716	0.633	0.58	0.61	0.67	0.61
Decision Tree	0.575	0.643	0.672	0.606	0.57	0.61	0.67	0.57
Adaboost	0.64	0.701	0.725	0.645	0.68	0.77	0.81	0.68
Gradient Boosting	0.637	0.715	0.744	0.627	0.67	0.78	0.81	0.7
Random Forest	0.656	0.732	0.801	0.67	0.68	0.8	0.88	0.74
Extremely Trees	0.656	0.71	0.811	0.69	**0.69**	0.79	**0.89**	0.73
Linear SVM	0.646	**0.782**	0.811	**0.712**	0.68	**0.84**	0.88	**0.75**
SVM	0.657	0.656	**0.817**	0.664	0.63	0.61	0.87	0.68
Average	***0.6325***	***0.685***	***0.7503***	***0.649***	***0.631***	***0.715***	***0.806***	***0.659***

fine tuning implementations. This is very discouraging since this was the model with better performance for feature extraction until now. Probably, it means that the deeper models will have even worse results than this. At the end, the theory and experiments analyzed at the related work can confirm that it is necessary train at least the half of the VGG19 network to obtain at least, similar results than a model trained from scratch.

ResNet152

ResNet architecture is a very deep network, for that reason filters are smaller and other filters like pooling, stride, batch normalization and activation functions for each specific block are implemented. Basically, batch normalization and activation function for each 'output' of one block for the following block input, makes possible with the residual implementation, that vanishing weight and learning leakage affects the model perform. Resnet152 is compound of 5 main layers blocks, at first block is the biggest filter, 7 x 7 using stride of 2, a 3 x 3 filter followed by a max pooling layer and stride 2. For the rest of blocks is pattern of 1 x 1, 3 x 3 and 1 x 1 filters followed by batch normalization and ReLu activation function. These filters are used different times according to the block. Block 2 uses 3 filters, block 3 uses 8, block 4 uses 36 and finally block 5 uses 3.

Distribution of filters by block has a high disproportion, at least for block 4 which contains 70% of the filters. Due this issue, is not possible to have a similar balance than VGG19, because we cannot unfreeze for example block 3 and block 5, there will be just an incoherence of weights at the activation function between blocks. Then, since train blocks 4 and 5 is not a real alternative of efficacy, block 5 was the only trainable block for this experiment.

Different from what was expected, on table 3.10 we can see that there was an improvement in ResNet152 model using fine tuning instead of transfer learning. In this case, it was a

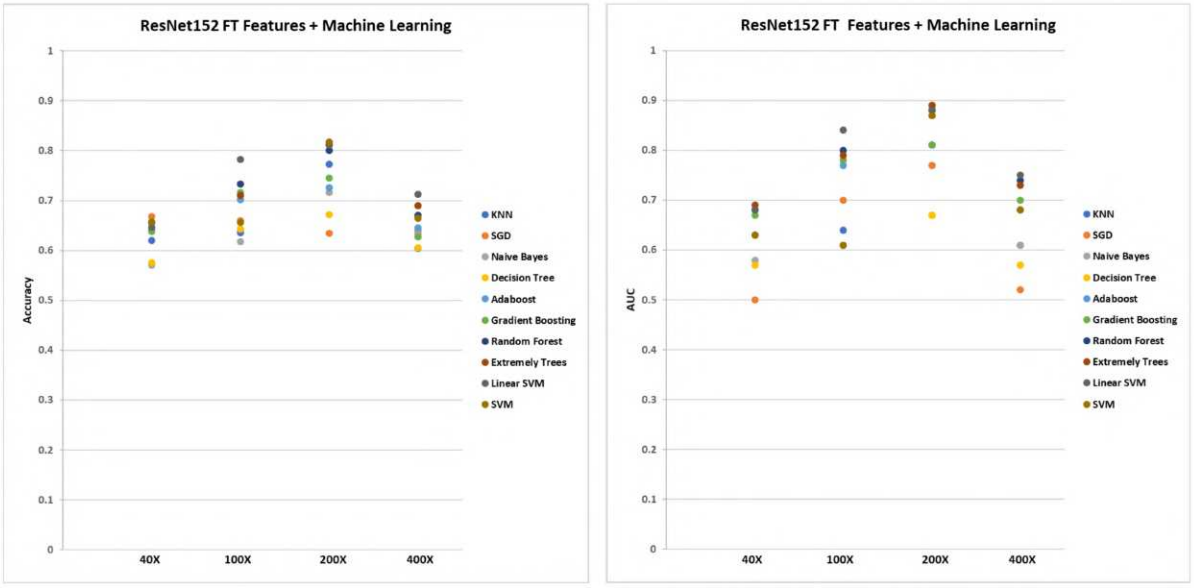

Figure 3.13: BreakHis machine learning performance using ResNet152 features with fine tuning.

curious performance, 40X is the magnification were results have been better until now, in this case was the worst result and the only magnification that was worse than previous implementation in fine tuning . The rest of magnifications obtain an improvement of 5% at 100X and 200X, the 400X magnification obtains an improvement of 2% in accuracy. That could mean that maybe with more trainable layers the results could be better, but we have seen that these blocks have high disproportion and adding another block will increase the trainable layers in more than 75%.

Inception V3

Inception v3 architecture contains 9 principals modules. Each module is characterized for the Inception quality, parallel filters and a concatenation filter that is the input for the next module. Each module contains 5 filters, 1 of size 1 x 1, 2 of size 3 x 3 and 2 filters of 5 x 5, followed by a pooling layer. Between each 3 modules there are average pooling, flatten, dense and dropout layers.

In this case, modules are well balanced, making possible a consistent combination of modules for fine tuning experimentation. In our approach we train the last module, which represents 12% of the model. We decide to implement this since even when there is not a big number of parameters, there is a big number of layers and filters, this is the deepest model for our project.

There is not any surprise at this model. Outcomes on table 3.16 were the same, that looks that for Inception network there only exist one class. This is the most complex model, and maybe the model which needs more resources and training. Of course does not mean that

Table 3.11: Accuracy and area under the curve for BreakHis dataset using Inception V3 fine tuning.

Technique	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.515	0.552	0.537	0.525	0.5	0.52	0.54	0.54
SGD	0.637	0.384	0.65	0.464	0.47	0.5	0.5	0.51
Naive Bayes	0.467	0.584	0.446	0.383	0.46	0.46	0.39	0.49
Decision Tree	0.493	0.54	0.556	0.554	0.49	0.51	0.52	0.53
Adaboost	0.491	0.536	0.567	0.519	0.49	0.47	0.57	0.54
Gradient Boosting	0.507	0.531	0.552	0.523	0.49	0.48	0.56	0.55
Random Forest	0.533	0.539	0.583	0.541	0.51	0.52	0.58	**0.56**
Extremely Trees	0.547	0.53	0.571	0.528	**0.52**	**0.52**	**0.58**	0.55
Linear SVM	0.469	0.522	0.532	0.525	0.43	0.45	0.51	0.52
SVM	**0.657**	**0.656**	**0.657**	**0.638**	0.47	0.53	0.53	0.47
Average	***0.5316***	***0.5374***	***0.5651***	***0.52***	***0.483***	***0.496***	***0.528***	***0.526***

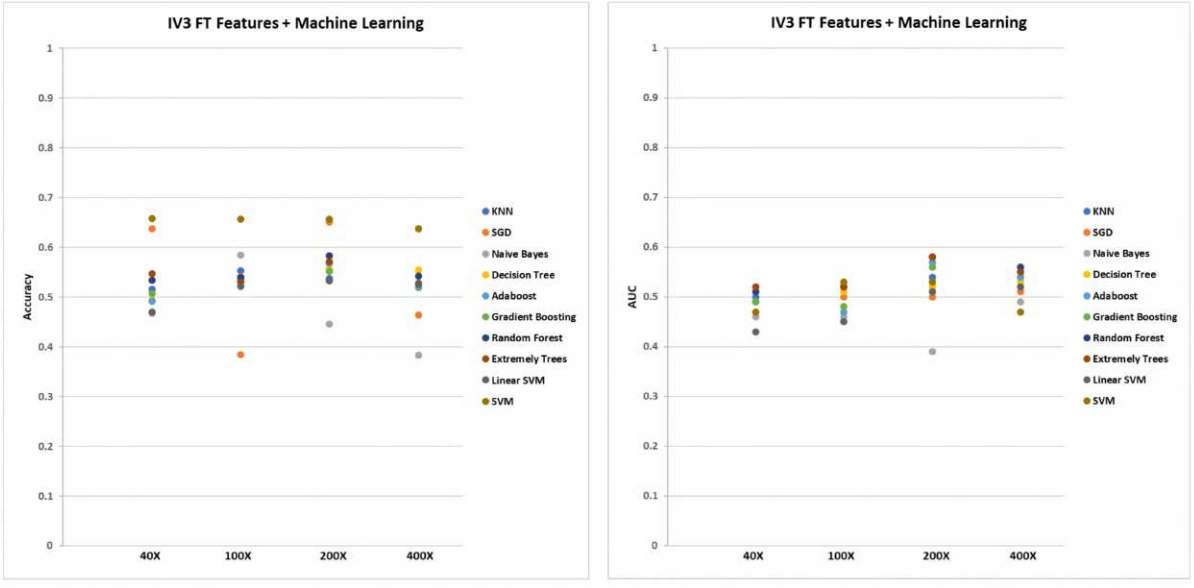

Figure 3.14: BreakHis machine learning performance using Inception V3 features with fine tuning.

Table 3.12: Accuracy and area under the curve for BreakHis dataset using VGG19 + ResNet152 fine tuning.

Technique	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.748	0.753	0.782	0.632	0.78	0.78	0.79	0.65
SGD	0.755	0.436	0.771	0.76	0.74	0.77	0.61	0.5
Naive Bayes	0.761	0.644	0.701	0.673	0.7	0.81	0.59	0.6
Decision Tree	0.72	0.701	0.732	0.682	0.71	0.69	0.71	0.66
Adaboost	0.765	0.731	0.791	0.725	0.81	0.8	0.84	0.76
Gradient Boosting	0.763	0.725	0.798	0.74	0.83	0.79	0.85	0.78
Random Forest	**0.798**	**0.785**	0.81	0.74	0.84	**0.85**	0.87	0.8
Extremely Trees	0.797	0.782	**0.814**	0.737	**0.85**	0.85	**0.87**	0.8
Linear SVM	0.766	0.768	0.791	**0.748**	0.8	0.84	0.85	**0.82**
SVM	0.657	0.656	0.657	0.722	0.5	0.65	0.53	0.77
Average	***0.753***	***0.6981***	***0.7647***	***0.7159***	***0.756***	***0.783***	***0.751***	***0.714***

it is a bad model, but his best performances is obtained when it is fully trained from scratch or at least that has been reported in related works and probed on these experiments. Qu demonstrate the same using VGG16, AlexNet and Inception V3 as architectures for his fine tuning experiments [31]. This has sense at that level that the VGG architecture is also our best model.

Following the related works, Kumar et al. uses the combined features from two different convolutional models [30]. Then, model combination using machine learning is not a crazy idea. Similar what we had done in transfer learning, we apply the same combinations for fine tuning. Outcomes were quite direct, the combination obtain the best results on table 3.15 from VGG19 what were magnifications 40x and 400x, and results at 100X and 200X magnification of ResNet152. In nutshells, the combination conserves the better outcomes of the models but it still being worse than the same combination using transfer learning. Similar than transfer learning, the full combination, on table 3.16, adding the Inception model does not improve anything, it just reduce at 1% the 40X magnification accuracy and increased 1% 400X magnification results. Finally, there is an important lesson obtained from this experiment, it will be really necessary to use a baseline model trained from scratch to achieve good outcomes on BreakHis dataset. We will need to use all our resources to train a deep model, even changing the machine learning techniques for other possible deep learning technique compatible with convolution networks.

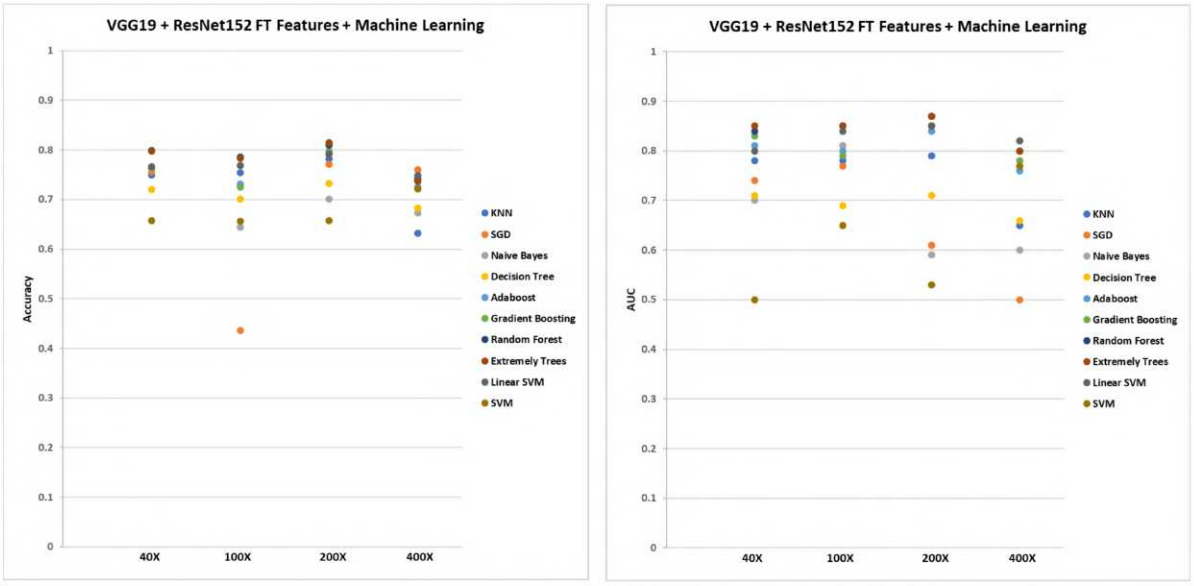

Figure 3.15: BreakHis machine learning performance using VGG19 + ResNet152 features with fine tuning.

Table 3.13: Accuracy and area under the curve for BreakHis dataset using VGG19 + ResNet 152 + Inception V3 fine tuning.

Technique	**Accuracy**				**AUC**			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
KNN	0.715	0.621	0.725	0.508	0.72	0.6	0.74	0.52
SGD	0.711	0.713	0.633	0.5	0.65	0.59	0.53	0.55
Naive Bayes	0.762	0.688	0.708	0.693	0.71	0.81	0.6	0.61
Decision Tree	0.7	0.69	0.74	0.648	0.68	0.68	0.74	0.63
Adaboost	0.77	0.734	0.809	0.722	0.82	0.78	0.86	0.76
Gradient Boosting	0.759	0.734	0.795	0.723	0.81	0.8	0.85	0.77
Random Forest	0.783	0.756	0.81	0.748	0.84	0.83	0.87	0.8
Extremely Trees	**0.787**	**0.768**	**0.814**	0.745	**0.84**	**0.84**	**0.88**	**0.8**
Linear SVM	0.748	0.75	0.793	**0.754**	0.79	0.81	0.83	0.79
SVM	0.657	0.656	0.657	0.638	0.53	0.5	0.47	0.53
Average	***0.7392***	***0.711***	***0.7484***	***0.6679***	***0.739***	***0.724***	***0.737***	***0.676***

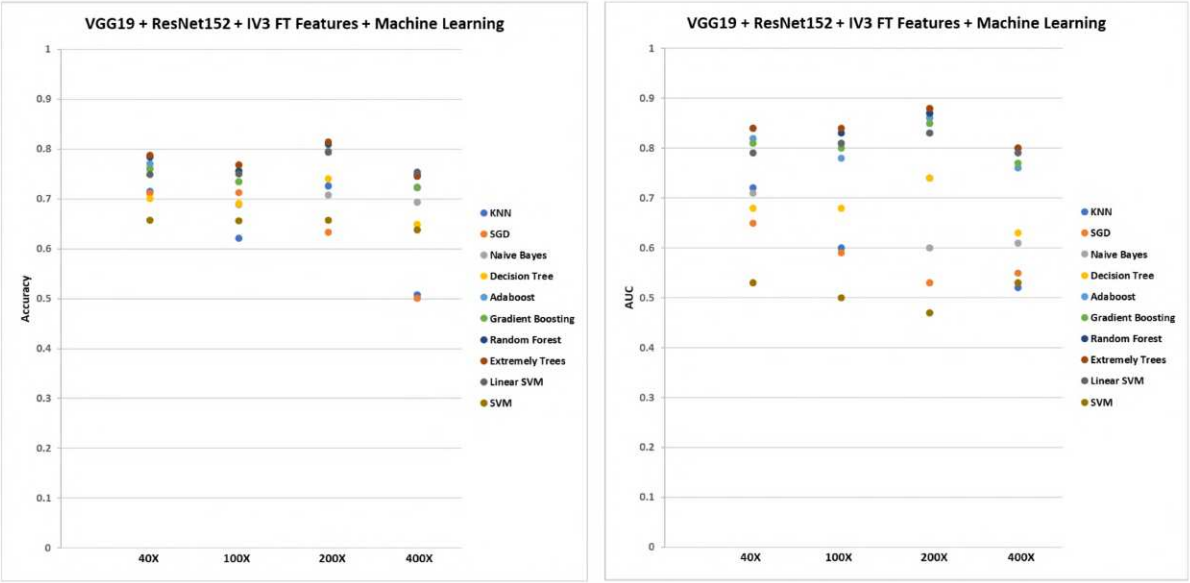

Figure 3.16: BreakHis machine learning performance using VGG19 + ResNet152 + Inception V3 features with fine tuning.

3.4 Convolutional Neural Networks + Recurrent Neural Networks

From previous experiments, partial conclusions indicated that baseline model was the best option for digital pathology classification. There were some similar approximations from transfer learning using VGG19 network. In the case of LC25000 dataset, results were better than using a baseline model. For BreakHis dataset, outcomes were not conclusive in any model and related work indicated that there is a very important improvement on outcomes for the same problem. Then, we have seen that using different networks from the simple one to the deepest model is not a great strategy. Then, we decide to add a new characteristic for a baseline model.

To compare a real alternative of deep learning before we need to express the real potential of deep learning for the same problem. For that conditions, we decide to not use machine learning as a complement of the model and use recurrent neural networks as a complement of the model. These networks are rarely used to image task, but in this case combined with a convolution neural network can perform a great improvement.

Following the scope of our project, this experiment cannot be performed with different combinations like architecture comparison. Even, for this occasion, patching will not be implemented by the number of images and data that could represent. The function of this experiment is to represent how far are the outcomes between deep learning and machine learning techniques using base line, transfer learning and fine tuning from a complex model compound by 2 different deep learning approaches.

3.4.1 Xception

For this implementation we decide to include a new architecture, that is practically an extreme version of Inception architecture. The selection of Xception network for this section was don for previous and partial outcomes obtained at BreakHis dataset using 200x magnification. Architecture was compared with ResNet152 and Inception V3 networks obtaining 3/% and 4/% better accuracy respectively after 5 iterations.

Xception model was published by François Chollet in 2017, just one year after the publication of GoogLeNet [32]. The premise of this model is modification of the depthwhise separable convolution at Inception, were the deptwhise convolution followed by pointwise convolution, this last convolution is the premise of Inception model, the 1 x 1convolution. Depthwise convolution is a channel spatial convolution of n x n. In our case our input is a 3 n x n RGB image. The main advantage of this convolutions is that they do not need to be executed in all the image channels.

The modification on Xception, is the order, pointwise convolution followed by a depthwise convolution. That means two changes, the first is the order of operations, 1 x 1 convolutions and after channel wise spatial convolution. Second is that there is not non-linearity, which means that there are not intermediate ReLu activation. This represents a parameter reduction of the model, 20 million compared with 23 million of Inception network.

3.4.2 Long Short Term Memory

Recurrent neural networks are similar to the tradition networks, but in this case, there are connections between nodes from a direct graph across a temporal sequence, giving the impression a temporal behavior. In 1997, Hochreiter and Schmidhuber present the long short term memory networks. Years after, the model was used at the beginning of a new area, deep learning. Speech recognition, predictive text, text creation and stock exchange predictions were just of the applications of these models. During the last decade, the use of the models has increased at lot due the boom of convolution networks, pattern recognition is probably the most common use of this networks.

Long short term memory networks are capable to learn long term dependencies. Recurrent networks were not efficient for these problems due the high cost of remember information for long periods. The main difference between traditional networks was that, long short networks has 4 neural layers. These layers are basically, the input of new information, input for the output of the previous cell, output of information and the output information for the next cell unit. Additionally, these networks have the capacity to remove or add information to the current state by some structures called gates. These are a way to let information and conserve just the important information. These gates are composed of a sigmoid layer and a pointwise multiplication operation.

Then, these networks are capable to conserve information, recognize patterns, characteristics and main components of the input features, does not mattering if there are long inputs like a textbook, long sequences of audio or pixels from high dimensional images. This model could be a perfect complement for the feature process doing by a convolution network.

3.4.3 Related Work

Budak et al. use a combination of convolutional neural networks and long short term memory networks for detection of breast cancer in histopathological images using the BreakHis dataset [34]. Authors propose an end-to-end model using a convolution network to encode the input image into a high level representation. The last flatten layer of the convolution networks is used as the input of bidirectional long short-term. For experimentation were selected 3 convolution networks, AlexNet, VGG and ResNet. Author separates the dataset randomly and then testing the different networks. Author obtains better results using AlexNet model, giving an accuracy of 95.69 for 40X magnification, 93.61 for 100X, 96.32 at 200x and 94.29 for 400x magnification. These outcomes are high close to the results obtained by Han et al., but still being not better. This is an indication that probably, it could be a good approximation.

Zainudin et al. performs a similar experiment using the AMIDA13 dataset [35]. For this work, the problem is the classification of samples with metastasis an no metastasis. Authors implement 4 possible models, multilayer perceptron, LeNet and AlexNet as baseline models and convolution network + long short term memory, using the flatten network as the input of the recurrent network. Outcomes obtained were very overwhelming, obtaining 85% of accuracy for the combination of convolution and recurrent networks, 63/% for LeNet, 54 /% for AlexNet and 41% for multilayer perceptron. Different for machine learning, recurrent networks look like a great boosting for convolution networks.

Finally, Dubey et al. uses the same combination for ischemic and non-ischemic cardiomyopathy histopathological images classification [36]. Authors use Inception V3 and long short term memory as the unique model implemented. Different combinations are implemented with these networks, Inception as the baseline, Inception + recurrent network and Inception + recurrent network + self attention mechanism. Results, were again better for the last model, obtaining 95.38% of accuracy, Inception + recurrent network obtains 93.84% and finally the convolution model as baseline obtained 92.30%. This has been encouraging, since Xception, the model selected for our experiment is the improved version of GoogLeNet and Inception v3, some of the used networks on these works.

3.4.4 Results

The model selected for experiments consist in Xception network tacking a 299x299x3 RGB image as input. On the convolution network, this image is passed through the pretrained Xception model with the ImageNet weights until it reaches the final convolution block, were the features are extracted, this is of size 2048. On the recurrent network, the image is transformed

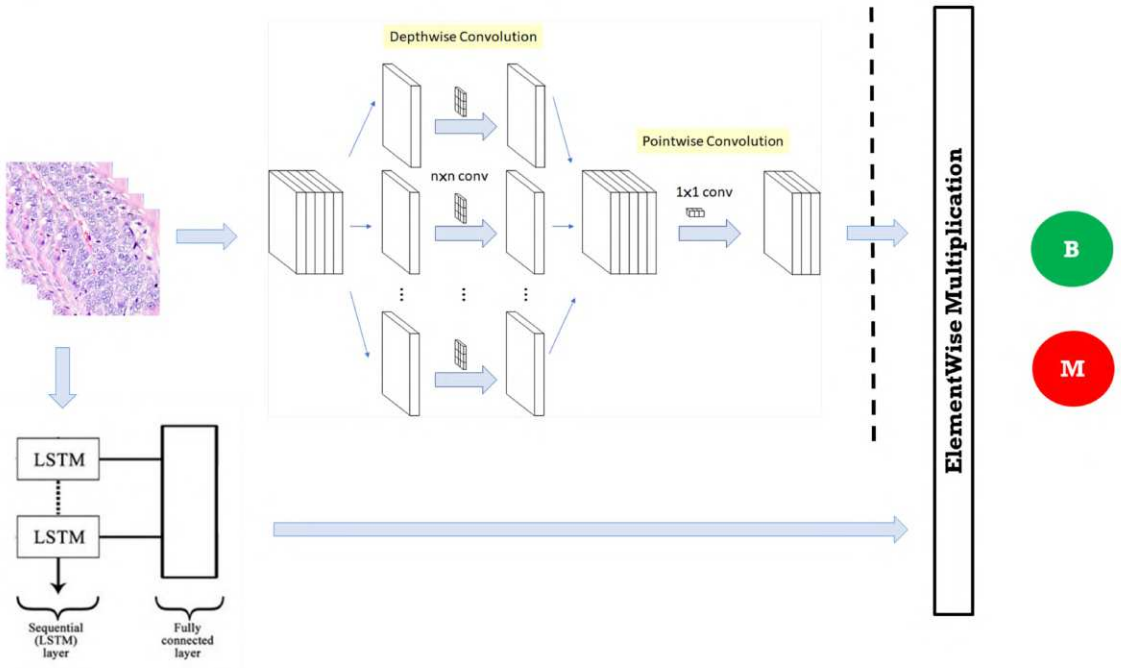

Figure 3.17: Convolutional Neural Networks + Recurrent Neural Net-works

to a gray scale image of size 299 x 299 x 1, since there just channel, to be able to split it and feed it into the recurrent network. Then, the image is reshaped into (23, 3887), where 23 is the timesteps and 3887 is the dim of each timestep. These values were chosen because $233887 \approx 299299$. The image is then passed through two LSTM layers, each of which are 2048 features output. Finally, the two outputs are merged using element-wise multiplication. The output of this multiplication is then fed to the classification layer which consists of 2 nodes and a softmax activation, all the process is explained on figure 3.17.

Network was trained in two phases. First, all the layers of the CNN were frozen and only the last classification layer and recurrent network were trained. This was done using the RMSProp optimizer. At the second phase, all the layers of the entire network were unfrozen and finetuned using Adam using the same hyperparameters than the ResNet152 patching experiment.

Outcomes of this strategy were the most accurate in our project, we can see the results on table 3.14. Even when the performance for LC25000 is almost perfect, in BreakHis dataset, were far for the obtained at the state of the art, at least for 200X and 400X magnification. For related works like Budak and Han et al. which have the better outcomes, we only have a difference of 3% and 2% in accuracy for 40x and 100X magnifications respectively. Different from the results obtained at accuracy, the area under the curve gets a great improvement, obtained an almost perfect performance excepting for the 400X magnification.

In nutshells, this is a high deep learning implementation that has need several hardware resources to being implemented, different from the convolution network baseline, having a

Table 3.14: Accuracy and area under the curve for convolutional neural network + recurrent neural network experiments.

Dataset	**Accuracy**	**AUC**
Colon	1	1
Lung	0.997	1
Breast 40X	0.92	0.98
Breast 100X	0.918	0.98
Breast 200X	0.8871	0.97
Breast 400X	0.884	0.92
Average	***0.934***	***0.975***

great improvement. Outcomes could imply that it will be hard to obtain similar results for techniques that do not include deep learning. This can be deducted when deep learning techniques that used less resources like fine tuning or transfer learning were not effective.

Chapter 4

Handcrafted Features

Image processing and handcrafted features are probably some of the branches that have been displaced by the continuous use of convolution networks. Deep learning for image classification, segmentation and other tasks that have given great results, but they still given the same black box issues since always. There are some convolution network architectures designed for specific targets like textures, face recognition and thousands more, but that does not evade the fact that the network could learn some different factor or pattern different from the desired, making the model unpredictable.

For that reason we decide to apply 3 handcrafted features based on textures; Haralick textures, Local Binary Patterns and Local Phase Quantization [37][38][39]. There were used 2 handcrafted features based on points of interest; Scale Invariant Feature Transform and Histogram of Oriented Gradient[40][41].

These are considered handcrafted features not because they are obtained by human criteria. These are considered handcrafted because it is not necessary to train a model to obtain them. Also, it is not necessary many images to define these features. It means, each image feature is independent from other possible images. More images do not increase the quality of the features, but of course, more images represent more features samples that can improve the performance of the selected model to make a prediction. Then, we can conclude that these features are easier to obtain since does not need a complex trained model to be generated. These features could be an attractive alternative for deep learning considering requirements and complexity, but also, they have a big limitation, are designed to obtain concrete information for images and not for specific problems like convolution networks. In nutshells, is the best alternative for deep learning, but maybe not a real alternative to obtain similar outcomes, but this is what we will try to demonstrate.

4.1 Related Work

Handcrafted features are not a popular approach to tackle image task. Even when there have been several works related to computer vision analysis, these have been displaced for other

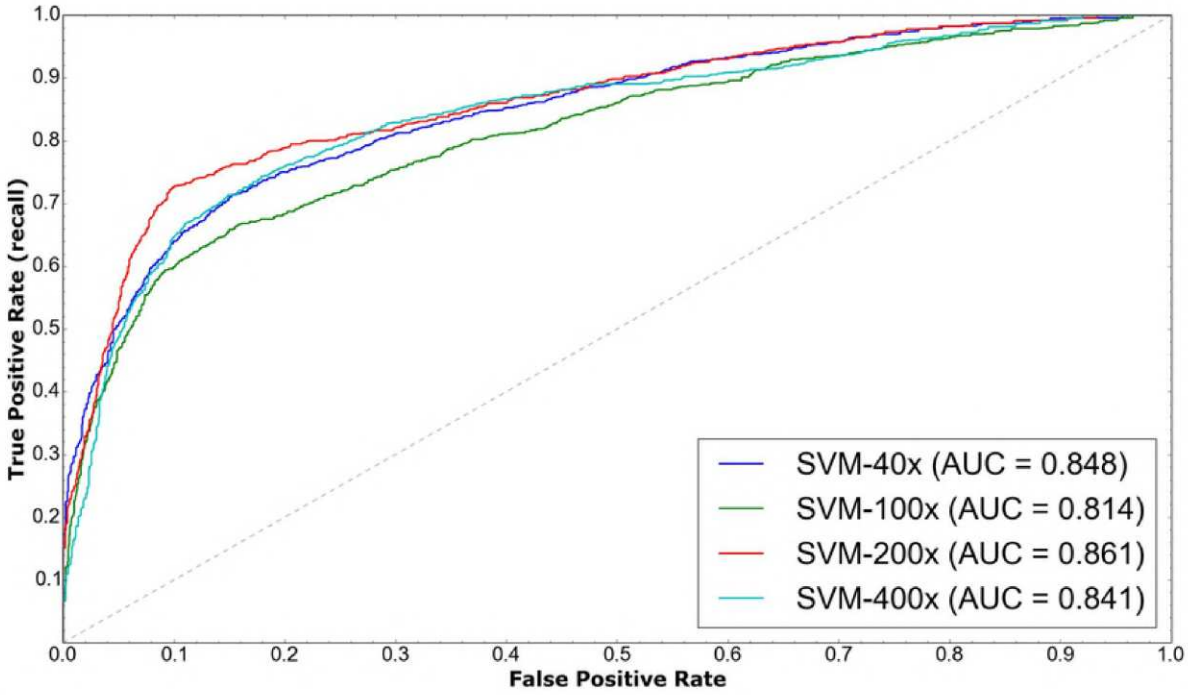

Figure 4.1: Area under the curve for BreakHis dataset using support vector machines and PFTAS features by Spanhol et al. [43].

approaches like deep learning. There are still being researches about these science field, but almost all of these works are focused in the improvement or development of new features. Most of the popular and effective features were developed 50 or 40 years ago. There are less works that applies these techniques and for consequence, few projects related with digital pathology or at least with the cancer digital pathology on which we are working.

Authors of BreakHis dataset, Spanhol et al., perform their own experiment using handcrafted features in the original dataset paper[43]. There were tested 6 different handcrafted features, Local Binary Patterns, Completed Local Binary Patterns, Local Phase Quantization, Haralick textures, Parameter Free Threshold Adjacency Statistics (PFTAS) and Oriented Fast and Rotated Brief (ORB). These features were classified using 1 nearest neighbor, quadratic linear analysis, support vector machine and random forest. Outcomes are better using support vector machines. PFTAS features is the feature with the best performance, followed by the Local Binary Pattern features. These outcomes are close to the presented at the deep learning implementation done by the same author, but still being not better.

Bardou et al. present a vert related work using different approximations for the classification of BreakHis dataset. [5]. In these experiments are used convolution networks and handcrafted features. Dense SIFT and SURF are the implemented features that were used to train a support vector machine. The outcomes in accuracy of SURF features were 79.95%, 74.80%, 70.96% and 72.02% for 40X, 100X, 200X and 400X magnifications respectively. In the case of Dense SIFT features the results were 52.68%, 56.22%, 50.12% and 50% for 40X, 100X, 200X and 400X magnifications. Applying different spatial pyramid labels, outcomes

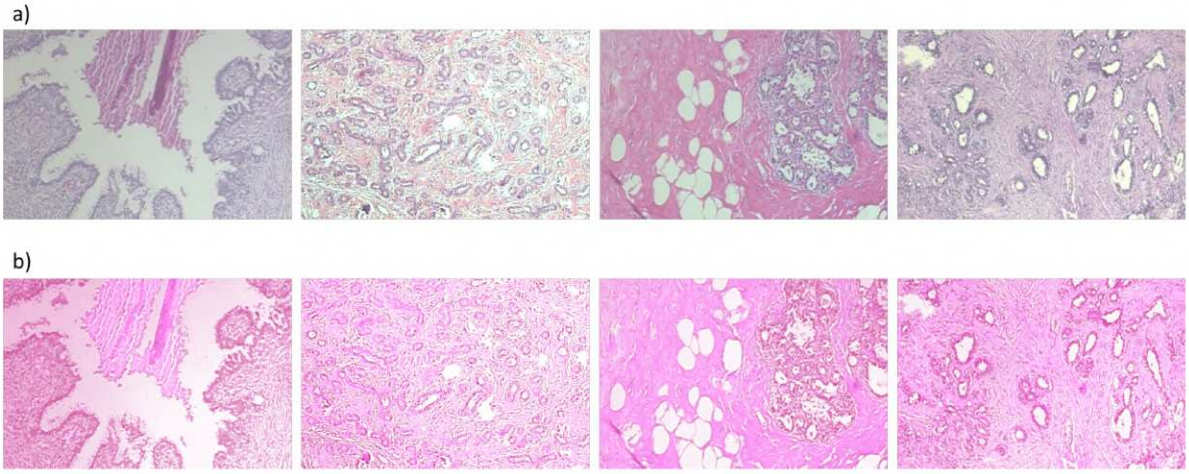

Figure 4.2: a) Original images from BreakHis dataset, b) Stain normalization for BreakHis dataset.

increased at 84.78%-85.79%, 81.57%-82.85%, 79.64%-83.77% and 82.05%-83.15% for 40x, 100X, 200X and 400X magnifications. The outcomes using the same technique with Dense SIFT features were 69.40%-70.06%, 70.67%-72.11%, 70.36%-71.19% and 67.77%-71.61%.

There are some handcrafted works related with colon digital pathology images. Li et al. present a colon histology images using convolutional neural networks and handcrafted features[44]. For the gland segmentation in colon histology authors used different handcrafted features, SIFT, Vectorized raw-pixel (RAW) and multi-resolution Local Patterns (mLP). A support vector machine model was trained with the different extracted features. AlexNet and GoogLeNet were used as baseline model and features extractors. Handcrafted features obtained similar segmentation results, obtaining 0.75 ± 0.11, 0.77 ± 0.08 and 0.77 ± 0.09 for SIFT, RAW and mLP features respectively. Convolution networks obtain better outcomes with 0.84 ± 0.10 and 0.82 ± 0.12 for AlexNet and GoogLeNet respectively, combined they obtained 0.87 ± 0.08. Finally, the best performance was obtained from the combination of all the handcrafted features and deep learning features, which was 0.87 ± 0.08. It is a real approximation of handcrafted features and how far is his performance from deep learning, but even when they are not better, can improve the general performance.

4.2 Stain Normalization

Before the feature extraction, we normalize the images. As we know, pixel normalization is common technique before the model fitting on convolution networks. Convolution models perform a deep processing image, channel by channel, applying and getting several filters. Different from deep learning techniques, handcrafted features are obtained generally using gray scale images. Single channel information is easy to process and includes information of the 3 channels. In this case is not enough the gray scale due the big difference between color intensities. Histopathological images have different color ranges, this is common due

the hematoxylin and eosin applied to the tissue. These substances highlight nucleus of cells and glands with blue color and coloring external components like cytoplasm with blue color. The quality and quantity of the hematoxylin and eosin, tissue reactions and digital scanning are some of the factors that produce a big diversity of coloration between all the samples.

Hematoxylin and eosin colors, also named stain colors, change from image to image and could be summarised in a RGB Stain matrix 4.1.

$$S = \begin{pmatrix} H_R & H_G & H_B \\ E_R & E_G & E_B \end{pmatrix} \tag{4.1}$$

A normal RGB image I is transformed to a RGB optical density image OD using the Beer Lambert law on equations 4.2 and 4.3.

$$I = 255 \cdot exp(-OD) \tag{4.2}$$

$$OD = -log(\frac{I}{255}) \tag{4.3}$$

Npix x 3 could be represented as the flatten OD image using the function 4.4. With Npix being the number of pixels. It is possible to relate the images OD array and the stain matrix S using pixel concentration matrix C (Npix x 2 array where columns are the pixel concentration of hematoxylin and eosin).

$$OD_{flat} = CXS \tag{4.4}$$

To normalize the color and then the gray scale intensity, we applied Stain normalization. This involves casting one image in the stain colors of a target image[42] decomposing the image into the stain matrix S and the concentration matrix C, then replace the stain matrix of the image to be transformed with that of the target image. Finally, we recombine them to give the final stain normalized image. The extracted features by handcrafted techniques were used to train the same 10 different machine learning techniques used on previous experiments.

Due the previous normalization of LC25000 dataset, Stain normalization was only applied to the BreakHis dataset.

4.3 Histogram of Oriented Gradients

In 2005 Dalal and Triggs present their work of HOG descriptors for human recognition [45]. The concept of these features is the description of intensity and edges using gradients. Input image is cropped into small regions called cells, each region size changes according to the images or problems characteristics. For the pixels of each cell, a histogram of gradient directions

is performed. The final features are the union of these histograms. Operating on cells make the calculation invariant to geometric and photometric transformations.

Main technique to calculate the gradients is the 1-D centered, point discrete derivative mask in vertical and horizontal directions. Gradient calculation requires filtering the intensity using the kernels [-1, 0, 1] and his transposed. Once obtained the cells, each pixel votes for a possible orientation according to his histogram value. Depending on the images, could be considered histograms until 180 or 360 grades, if there is not any orientation for the image or if it is an important factor. Gradient strengths need to be locally normalized; this can be done by grouping cells. Then the HOG features of an image are component vector of normalized cell histograms from the blocks. Overlapping and block size are important factors than can improve the final outcomes. Finally, these blocks are normalized, using v, the non normalized vector with all the histograms of the block, and e, a constant. Normalization $L2$ can be performed using equation 4.5 and $L1$ using equation 4.6. An additional normalization using the square root could be implemented using function 4.7.

$$f = \frac{v}{\sqrt{||v||_2^2 + e^2}} \tag{4.5}$$

$$f = \frac{v}{||v||_1 + e} \tag{4.6}$$

$$f = \sqrt{\frac{v}{||v||_1 + e}} \tag{4.7}$$

4.4 Scale Invariant Feature Transformation

Scale invariant feature transformation (SIFT) is a concept published by Lowe in 1999 [46]. The main objective of this analysis is the extraction of key points in an image, then with these points, a possible object could be recognized on different images comparing features. Key point extraction represents a great advantage since an object match could be done does not mattering scales or rotations.

SIFT process is compounded by different steps. First, scale space is performed by the function $L(x, y, \sigma)$ by Gaussian convolution. Space is divided by octaves according to the size of the image, then, each octave is bulled using Gaussian blurring 4.8. Once obtained the blurred images, is implemented the difference of Gaussians, this process is just the difference between two Gaussian blurred images using different σ's like equations 4.8 and 4.9. Once obtained the difference of Gaussian, that differences are used to calculate Laplacian of Gaussian which is just an invariant scale. This step just consists in the comparison of one pixel with 8 neighbor's pixels, and 9 in previous and next scale. This step has generated many key points. To reduce this, is applied a technique to reduce edge features, which has not use full data due the image contrast. To detect these is used a Taylor series of scale to obtain locations and intensity of a threshold, the author recommends 0.03.

$$L(x, y, \sigma) = G(x, y, \sigma)I(x, y) \tag{4.8}$$

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-x^{(2}+y^2)}{2\sigma^2}} \tag{4.9}$$

The last steps are the orientation and key points description. For orientation we select a key point and the pixels in the neighborhood are used to calculate the magnitude and direction. Once calculated the gradients for all the pixels around the point will be a peak on the histogram. This peak is used to obtain the object orientation. Once obtained orientation, the location and scale, the features are finally obtained the descriptor each key point. Main implementations consist in a 16 x 16 windows around the key points and divided each window at 16 blocks. The histogram is calculated for each window, and the descriptor is obtained by the 16 blocks. These histogram orientations are independent of possible features orientations at the images.

4.5 Haralick Textures

Gray level co-occurrence matrix is a method proposed by Haralick et al. in 1973 [46]. This technique could be the easiest and more direct feature. The only pre-processing technique applied is a gray scale transform. Haralick propose a technique capable to show how often a gray intensity occurs in a geometric way or position between some pixels. According to the problem if orientation matters, 0, 45, 90 and 135 grades are the main orientations to compare this gray patterns.

An offset ($\Delta\boldsymbol{x}$,$\Delta\boldsymbol{y}$) is the position operator for all the pixels in the image. If an image contains $\boldsymbol{p}$ pixels values, the co-occurrence matrix will have a $\boldsymbol{p}$ x $\boldsymbol{p}$ dimension. Finally, The co-occurrence matrix $\boldsymbol{C}$ of an image $\boldsymbol{I}$ can be defined as fallows.

$$C_{\Delta x,\Delta y}(i, j) = \sum_{x=1}^{n} \sum_{y=1}^{m} \begin{cases} 1, & \text{if I(x,y) = i, and I(x + } \Delta_x\text{, y + } \Delta_y\text{) = j} \\ 0, & \text{otherwise} \end{cases}$$

Where i and j are the pixel values an x and y the spatial positions.

4.6 Local Binary Patterns

Described by first time in 1994 by Ojala et al., these features patterns are visual features used for image classification [48]. Process to obtain these features is similar to the histogram of oriented gradients. Basically, it consists in dividing the image in cells, according to the size of the image could be more or less cells. Each pixel is compared to each other among his 8 different neighbors. If the center pixel is bigger than the neighbor the value is 0, if not it is 1 as is expressed on equations 4.10 and 4.11. This is generally calculated using the gray scale images. The histogram of each cell is calculated using the frequency of each value to generate

the feature vector of the image.

$$H_i = \sum_{x,y} I f_1(x,y) = i, i = 0, ..., n-1 \tag{4.10}$$

$$N_i = \frac{H_1}{\sum_{j=0}^{n-1} H_j} \tag{4.11}$$

4.7 Local Phase Quantization

One of the current studies with more impact at the image processing area is the local phase quantization. In 2008 Ojansivu et al. present local phase quantization for texture classification [49]. Like the rest of features, image blurring is doing as part of the pre-processing. One of the main components of this descriptor is the short-term Fourier transform. Other similar component is the statistical analysis of coefficients. Assuming that $f(x)$ is the output of Markov process using a correlation coefficient between each pixel. Considering (σ^2= 1), covariance can be defined as equations 4.12 and 4.13, where ($||\cdot||$) is the L_2 norm for all the M samples in N_x.

$$\sigma_{i,i} = p^{||x_i - y_j||} \tag{4.12}$$

$$C = \begin{pmatrix} 1 & \sigma_{12} & \cdots & \sigma_{1M} \\ \sigma_{21} & 1 & \cdots & \sigma_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{M1} & \sigma_{M2} & \cdots & 1 \end{pmatrix} \tag{4.13}$$

Finally, the covariance matrix of the transform coefficient vector F_x is obtained as equation 4.14.

$$D = WCW^T \tag{4.14}$$

Last step before the quantization is a de-correlation of coefficients. The objective of this step to preserve information if the images comes not from the same distribution. Using the Gaussian distribution, the independence is calculated using equation 4.15 with V as the orthonormal matrix from the decomposition of D matrix on equation 4.16.

Table 4.1: Accuracy for LC25000 dataset using handcrafted features.

Technique	Accuracy							
	HOG		SIFT		Haralick		LBP	
	Lung	Colon	Lung	Colon	Lung	Colon	Lung	Colon
KNN	0.406	0.53	0.287	0.491	0.863	0.744	0.948	0.971
SGD	0.729	0.633	0.238	0.532	0.733	0.658	0.6977	0.748
Naive Bayes	0.7	0.613	0.3	0.503	0.649	0.571	0.776	0.82
Decision Tree	0.618	0.634	0.299	0.484	0.86	0.763	0.934	0.962
Adaboost	0.761	0.696	0.262	0.525	0.624	0.69	0.831	0.968
Gradient Boosting	0.786	0.722	0.261	**0.533**	0.853	0.705	0.932	0.976
Random Forest	0.839	**0.767**	0.287	0.5	0.905	0.816	0.976	0.983
Extremely Trees	**0.839**	0.756	0.276	0.53	**0.906**	**0.825**	**0.986**	**0.988**
Linear SVM	0.798	0.675	0.302	0.531	0.882	0.79	0.902	0.967
SVM	0.794	0.686	**0.309**	0.528	0.868	0.779	0.909	0.964
Average	***0.732***	***0.672***	***0.28***	***0.514***	***0.818***	***0.735***	***0.893***	***0.938***

$$G_x = V^T F_x \tag{4.15}$$

$$D = U \Sigma V^T \tag{4.16}$$

G_x is calculated for all the image points and the outcome vectors are quantizied like local binary patterns as follows.

$$q_j = \begin{cases} 1, & \text{if } g_j \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

Where g_j is the jth element of G_x. Finally, the quantization of coefficients are integer values between 0 and 255 using binary representation using equation 4.17.

$$b = \sum_{j=1}^{8} q_j^{2^{j-1}} \tag{4.17}$$

Final representation is a histogram of these integer values of all the possible image positions, using 256 dimensions feature vector.

4.8 Results

Table 4.2: Area under the curve for LC25000 dataset using handcrafted features.

Technique	AUC							
	HOG		SIFT		Haralick		LBP	
	Lung	Colon	Lung	Colon	Lung	Colon	Lung	Colon
KNN	0.62	0.65	**0.45**	0.47	0.95	0.8	0.99	0.99
SGD	0.89	0.7	0.42	0.54	0.91	0.72	0.86	0.83
Naive Bayes	0.82	0.7	0.4	0.49	0.86	0.65	0.93	0.91
Decision Tree	0.72	0.63	0.46	0.49	0.89	0.76	0.95	0.96
Adaboost	0.89	0.76	0.39	0.52	0.96	0.77	0.98	1
Gradient Boosting	0.89	0.79	0.39	**0.54**	0.96	0.78	0.98	1
Random Forest	0.95	**0.85**	0.41	0.52	0.98	0.91	1	1
Extremely Trees	**0.95**	0.85	0.39	0.52	**0.99**	**0.91**	**1**	**1**
Linear SVM	0.91	0.73	0.35	0.53	0.96	0.84	0.96	0.99
SVM	0.94	0.75	0.39	0.53	0.97	0.84	0.98	0.99
Average	***0.858***	***0.741***	***0.405***	***0.515***	***0.943***	***0.798***	***0.963***	***0.967***

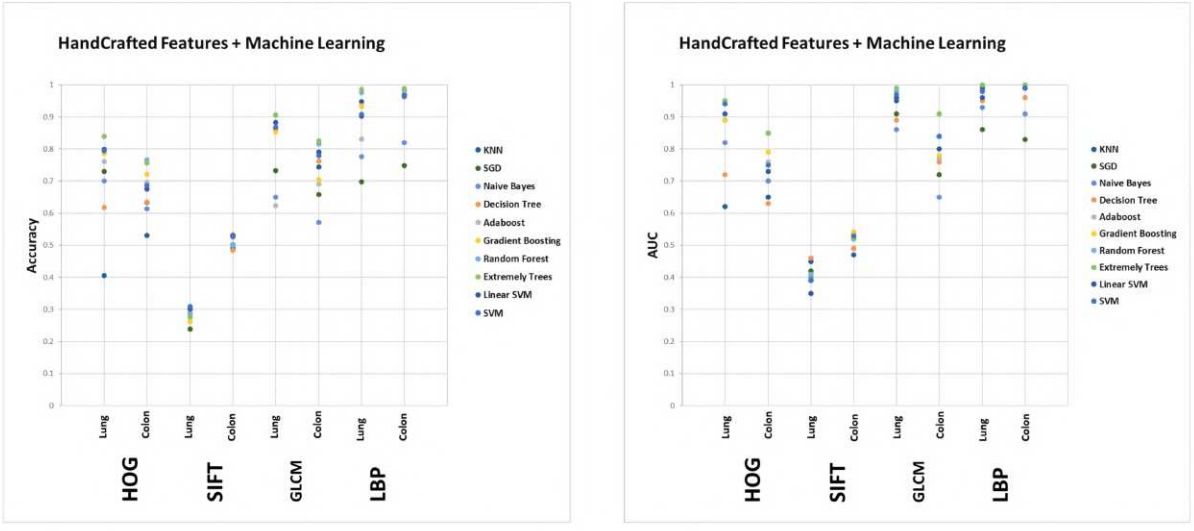

Figure 4.3: LC25000 dataset machine learning performance using handcrafted features.

Table 4.3: Accuracy and area under the curve for BreakHis dataset using handcrafted features.

Technique	HOG							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.559	0.55	0.404	0.387	0.49	0.53	0.6	0.58
SGD	0.595	0.406	0.529	0.609	0.5	0.55	0.58	0.61
Naive Bayes	0.583	0.515	0.557	0.587	0.48	0.41	0.53	0.7
Decision Tree	0.524	0.535	0.518	0.564	0.48	0.53	0.51	0.54
Adaboost	0.563	0.576	0.556	0.590	0.52	0.53	0.56	0.62
Gradient Boosting	0.531	0.567	0.571	0.595	0.46	0.54	0.58	0.63
Random Forest	0.542	0.538	0.571	0.629	0.51	0.53	0.56	0.69
Extremely Trees	0.550	0.546	0.59	**0.645**	0.51	0.54	0.56	**0.69**
Linear SVM	0.604	0.592	0.594	0.589	**0.51**	0.52	**0.56**	0.65
SVM	**0.657**	**0.656**	**0.657**	0.638	0.49	**0.54**	0.53	0.64

Technique	SIFT							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.714	0.589	0.591	0.583	0.72	0.56	0.58	0.59
SGD	0.621	0.403	0.466	0.564	0.69	0.51	0.54	0.56
Naive Bayes	0.624	0.522	0.517	0.548	0.66	0.5	0.51	0.56
Decision Tree	0.632	0.606	0.602	0.578	0.6	0.57	0.53	0.54
Adaboost	0.707	0.589	0.595	0.564	0.74	0.57	0.59	0.56
Gradient Boosting	0.702	0.592	0.604	0.6	0.74	0.56	0.61	0.61
Random Forest	0.683	0.589	0.608	0.574	0.72	0.58	0.59	0.58
Extremely Trees	0.692	0.614	0.6	0.583	0.75	**0.59**	**0.61**	**0.61**
Linear SVM	**0.716**	**0.656**	**0.657**	**0.638**	0.74	0.49	0.51	0.56
SVM	0.703	0.656	0.657	0.638	**0.75**	0.49	0.51	0.56

Technique	Haralick							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.613	0.598	0.610	0.587	0.62	0.6	0.64	0.59
SGD	0.633	0.571	0.634	0.459	0.5	0.55	0.68	0.71
Naive Bayes	0.413	0.576	0.583	0.445	0.54	0.52	0.7	0.71
Decision Tree	0.598	0.577	0.606	0.569	0.57	0.55	0.61	0.56
Adaboost	0.621	0.635	0.651	0.633	0.62	0.64	0.7	0.66
Gradient Boosting	0.65	0.631	0.653	0.645	0.64	0.66	0.7	0.68
Random Forest	0.644	0.640	0.641	0.642	0.68	0.67	0.7	0.66
Extremely Trees	0.649	0.636	0.630	0.636	**0.7**	0.65	0.7	0.68
Linear SVM	**0.657**	0.694	**0.708**	0.725	0.45	0.73	**0.76**	0.74
SVM	0.657	**0.709**	0.696	**0.732**	0.5	**0.74**	0.75	**0.76**

Technique	LBP							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.775	0.717	0.733	0.711	0.81	0.77	0.78	0.76
SGD	0.657	0.372	0.422	0.677	0.5	0.79	0.73	0.74
Naive Bayes	0.646	0.459	0.514	0.572	0.72	0.73	0.74	0.76
Decision Tree	0.718	0.713	0.706	0.723	0.71	0.69	0.7	0.72
Adaboost	0.782	0.738	0.723	0.758	0.86	0.78	0.81	0.84
Gradient Boosting	0.786	0.761	0.744	0.761	0.85	0.82	0.82	0.84
Random Forest	0.801	**0.786**	0.779	0.770	0.88	0.84	0.86	0.85
Extremely Trees	**0.805**	0.776	0.778	0.766	**0.88**	**0.85**	**0.87**	0.85
Linear SVM	0.789	0.767	**0.806**	0.806	0.84	0.82	0.86	0.88
SVM	0.797	0.76	0.806	**0.807**	0.84	0.83	0.87	**0.88**

Technique	LPQ							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.759	0.755	0.762	0.728	0.8	0.77	0.78	0.75
SGD	0.657	0.588	0.36	0.638	0.72	0.66	0.72	0.8
Naive Bayes	0.668	0.664	0.646	0.621	0.56	0.53	0.53	0.58
Decision Tree	0.736	0.718	0.717	0.687	0.71	0.68	0.69	0.63
Adaboost	0.789	0.767	0.766	0.775	0.85	0.82	0.82	0.82
Gradient Boosting	0.777	**0.775**	0.7486	**0.777**	0.84	**0.84**	0.81	**0.84**
Random Forest	0.805	0.755	0.767	0.738	0.88	0.83	0.85	0.8
Extremely Trees	0.8	0.739	0.748	0.711	0.88	0.82	0.84	0.79
Linear SVM	**0.821**	0.765	**0.787**	0.774	**0.89**	0.83	**0.84**	0.84
SVM	0.707	0.668	0.685	0.708	0.75	0.74	0.75	0.82

Technique	LBP + LPQ							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.747	0.752	0.784	0.735	0.76	0.78	0.82	0.77
SGD	0.655	0.678	0.38	0.702	0.67	0.79	0.77	0.81
Naive Bayes	0.664	0.668	0.658	0.639	0.55	0.53	0.56	0.59
Decision Tree	0.787	0.711	0.748	0.709	0.72	0.67	0.73	0.67
Adaboost	0.812	0.802	0.819	0.798	0.9	0.87	0.89	0.88
Gradient Boosting	0.837	0.790	0.818	0.815	0.92	0.87	0.89	0.89
Random Forest	0.828	0.776	0.787	0.755	0.89	0.83	0.87	0.83
Extremely Trees	0.825	0.757	0.768	0.723	0.88	0.84	0.85	0.82
Linear SVM	**0.859**	**0.830**	**0.862**	**0.854**	**0.94**	**0.9**	**0.91**	**0.92**
SVM	0.750	0.756	0.854	0.829	0.86	0.87	0.91	0.91

Technique	Haralick + LBP +LPQ							
	Accuracy				AUC			
	40X	100X	200X	400X	40X	100X	200X	400X
KNN	0.714	0.6513	0.6451	0.609	0.68	0.64	0.66	0.63
SGD	0.539	0.648	0.712	0.526	0.61	0.69	0.75	0.77
Naive Bayes	0.669	0.667	0.663	0.653	0.56	0.53	0.58	0.6
Decision Tree	0.782	0.701	0.763	0.703	0.74	0.68	0.73	0.67
Adaboost	0.838	0.788	0.829	0.795	0.91	0.84	0.89	0.88
Gradient Boosting	0.828	0.814	0.815	0.815	0.9	0.88	0.89	0.9
Random Forest	0.838	0.763	0.784	0.777	0.89	0.82	0.87	0.83
Extremely Trees	0.820	0.751	0.776	0.749	0.88	0.83	0.85	0.82
Linear SVM	**0.859**	**0.823**	0.857	**0.833**	**0.94**	**0.89**	**0.91**	**0.9**
SVM	0.777	0.771	**0.862**	0.810	0.86	0.86	0.9	0.9

Table 4.4: LC25000 dataset best outcomes by machine learning techniques using handcrafted features

Feature	Accuracy		AUC	
	Lung	Colon	Lung	Colon
HOG	0.839	0.767	0.95	0.85
SIFT	0.309	0.533	0.45	0.54
Haralick	0.906	0.825	0.99	0.91
LBP	**0.986**	**0.988**	**1**	**1**

Outcomes were quite interesting, since we can obtain a good performance using single handcrafted features. Local binary patterns were the features with better performance in both datasets. In LC25000, as we can see on tables 4.1 and 4.2, LBP achieve a perfect classification, in the case of BreakHis, was necessary include an extra feature, local phase quantization to achieve better results. Partial conclusions are of course centralized in the great efficiency of handcrafted features. In lung and colon images, Haralick textures and local binary patterns were the features with the higher accuracy and area under the curve, followed by the histogram of oriented gradients which have not been close of these features, but obtained a good result. In the case of breast images, the history was similar, the feature with the best result was local binary pattern, followed by local phase quantization and Haralick textures.

Looking for a pattern in these experiments, we can see that the features oriented to the detection of textures were the best ranked, by the other side, histogram of oriented gradients and scale invariant feature transformation, features oriented to the object detection and points of interest were the worst ranked. This has sense, since in histopathological images is hard to find a point of interest o particular fix structure present in one class. All the images are more related with patterns of cells and glands around several forms of tissue. That reason can explain the outcomes of both types of features.

In general analysis, handcrafted features satisfied the expected outcomes. First, because in LC25000 dataset they obtained a perfect outcome. In BreakHis dataset, which is the harder dataset on this project, the single local binary pattern obtained better area under the curve than the author of the dataset. Applying the same technique that we have observed in the state of art and in our previous experiments, we combine the best ranked features, and the performances increased, in this case local binary patterns and local phase quantization were the optimal combination. This technique generated areas under the curve higher than 90% as we can see on tables 4.3 and 4.5. We can conclude that handcrafted features are a real alternative for deep learning, of course they are not comparable to high level techniques like convolution networks with recurrent networks, but obtained a similar, and even better outcomes than convolutional networks. This, using less resources and time, than the used for deep learning techniques.

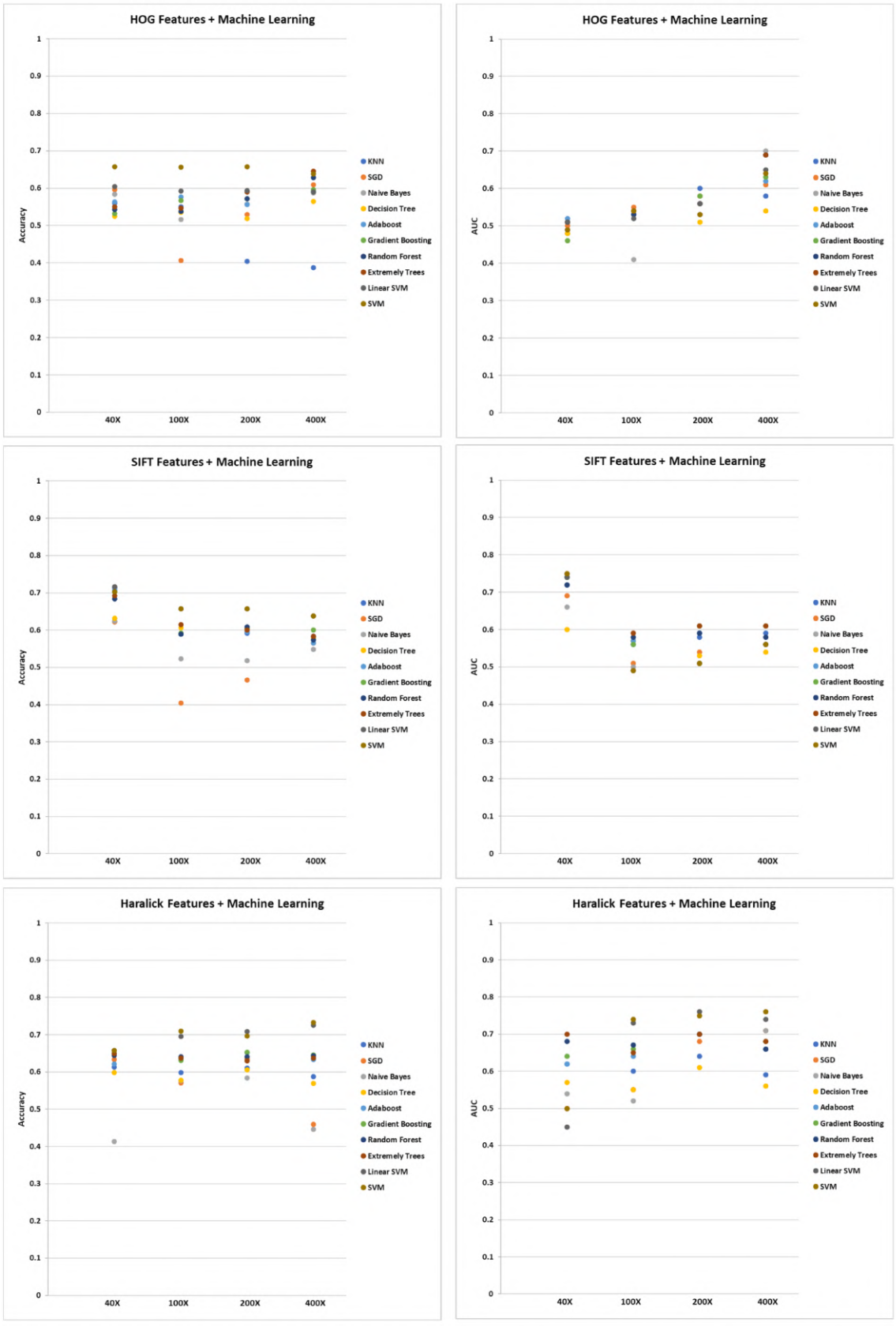

Figure 4.4: BreakHis dataset machine learning performance using handcrafted features.

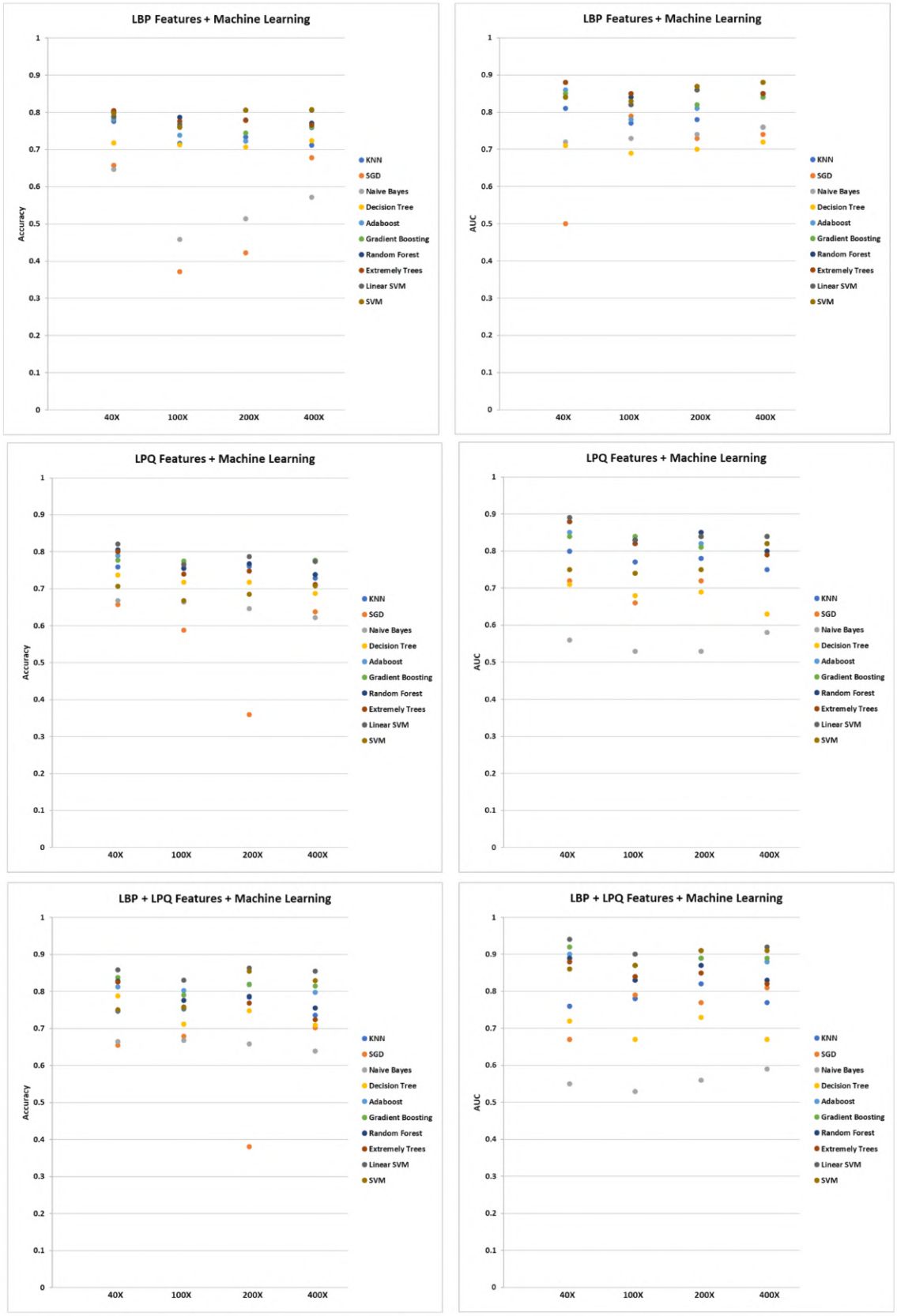

Figure 4.5: BreakHis dataset machine learning performance using handcrafted features.

Table 4.5: BreakHis dataset best outcomes by machine learning techniques using handcrafted features. Average of each feature is calculated using all the machine learning techniques.

Feature	Accurracy					AUC				
	40X	**100X**	**200X**	**400X**	***Average***	**40X**	**100X**	**200X**	**400X**	***Average***
HOG	0.657	0.656	0.657	0.638	***0.564***	0.52	0.54	0.6	0.7	***0.552***
SIFT	0.716	0.606	0.657	0.638	***0.609***	0.75	0.58	0.61	0.61	***0.596***
Haralick	0.657	0.709	0.708	0.732	***0.622***	0.7	0.74	0.76	0.76	***0.645***
LBP	0.805	0.786	0.806	0.807	***0.719***	0.88	0.85	0.87	0.88	***0.799***
LPQ	0.821	0.767	0.787	0.777	***0.721***	0.89	0.84	0.85	0.84	***0.767***
LBP + LPQ	**0.859**	**0.83**	**0.862**	**0.854**	***0.758***	**0.94**	**0.9**	**0.91**	**0.92**	***0.808***
LBP + LPQ + Haralick	0.859	0.823	0.862	0.833	***0.75***	0.94	0.89	0.91	0.9	***0.789***

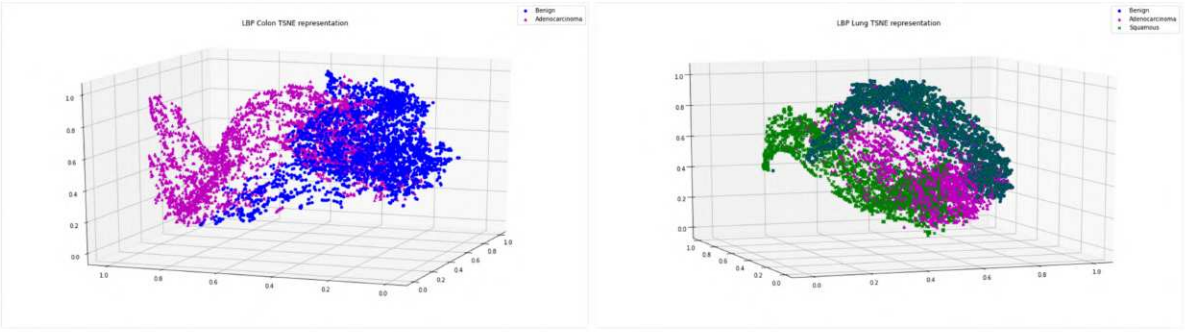

Figure 4.6: LC25000 Handcrafted T-SNE representation.

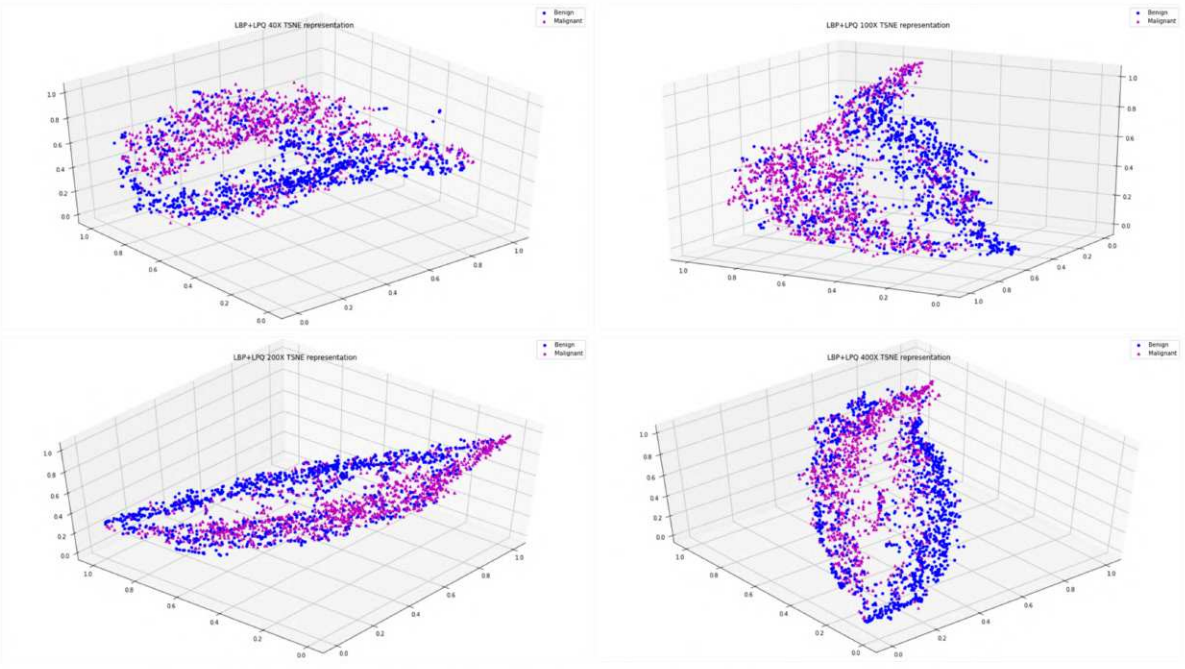

Figure 4.7: BreakHis Handcrafted T-SNE representation.

4.9 Models Ranking

On this project have been implemented 792 machine learning and deep learning techniques to tackle LC25000 and BreakHis datasets. These techniques contain different combinations and alterations of the most popular algorithms used to image classification. All the experiments, metrics and inference times were calculated used the following hardware characteristics.

- Intel(R) Xeon(R) CPU @ 2.30GHz
- 25 GB RAM
- Tesla P100-PCIE-16GB

Software requirements.

- Linux
- PyTorch 1.6
- Cuda 10.2
- Python 3.6
- imageio 2.9.0
- NumPy 1.16
- pandas 1.1.2
- PIL 7.2.0
- OpenCV 4.3.0
- scikit-image 0.16.1
- scikit-learn 0.22.2
- SciPy 1.19.0

This project generated 1,143 GB of information, containing Keras models obtained by hundreds of training iterations, image patches and normalized images. More than 500,000 images were generated for experimentation and almost 500 graphics of area under the curve, confusion matrix and T-SNE representations were generated.

General ranking by accuracy, area under the curve, inference time and other metrics like precision, recall and F1 score of the 792 techniques applied to the project can be obtained on the following source.

Models Ranking and Metrics

Chapter 5

Discussion

In this project have been implemented almost 800 techniques for histopathological image classification including deep learning, machine learning and computer vision. Partial conclusions were discussed on each chapter for the specific technique group. There was a great diversity on these implementations, great implementations with amazing evaluations and solutions based on these great implementations but worse evaluations. There is a possible explanation for all these cases, that will be discussed on the following sections.

5.1 Problem Strategy

At the beginning of this document we have mentioned that pathological classification has been a complex task for both, pathologists and artificial intelligence models. The main and more usually points to consider the complex level of this task are slide dimensions, color distribution, magnification factor, and class composition. In our first experiment we try to recreate a robust model with a technique which is very common for whole slide classification. Usually, patch generation is very helpfully wen the objective of the task is to compare tissue annotation. On the whole slides, pathologist use to 'sketch' the most interesting areas of the tissue according to the possibility of being a tissue. To perform this task by an artificial model, patching is very helpful, each patch is represented by its class probability over the whole slide. At the end we have a colorful map of cancer presence on the original slide. In our case that was not our intention, since we were looking for a metric comparison of different techniques, using a patching metric does not give us a comparable metric. The area under the curve obtained by patching technique was practically useless by due the almost binary classification of 0 and 1 after voting, applying the threshold and ignoring the rest of the probabilities.

We cannot say that patching is not a good technique for problems that does not imply whole slides or annotations. It has been proved on different projects that this technique could improve the classification task. The main question is in which problems this technique could be appropriated. Usually, all the convolutional networks have an input size smaller than 300 x 300 pixels, and these resize factor is commonly used even in images bigger than 1000 x 1000 pixels. If we are working with images with higher dimensions than 1000 square pixels, patching sound like a great idea, but if we are working with images of 768 x 468 and we will

transformed them in small images of 224 x 224 with a high overlapping factor, it could sound even ridiculous. Fortunately, there was an improvement, on partial experiments to select the best architecture, ResNet152 trained for the 200X magnification factor does not improve from 84% of accuracy after some iterations. We can not conclude that there will be always an improvement applying patching, but we can discuss if generating thousands and thousands of images to improve 3% or 4% of accuracy in the best case is a high cost or not. Of course, any improvement is always a good alternative, but we have seen that there are a lot of alternative to achieve this.

Data preprocessing is probably one of the most important factors for image classification. In our case we have to datasets with high contrasted characteristics, first, LC2500 is a preprocessed dataset, with two specific classes and great definition, BreakHis dataset is not a preprocessed dataset, with a proportion of 2:1 in the number of images between classes, each class is compound by several subtypes of cancer and all the image has the same image dimension, even when there are 4 different magnification factors, indicating that the resolution is not the optimal and information could have been lost in the resizing. If we compare the characteristics of both datasets is clear why the LC2500 dataset achieves great outcomes in almost all the techniques and why BreakHis gets good accuracy using just specific techniques, in specific the most complex. For example, without Stain normalization, the BreakHis dataset obtains no more than 73% of accuracy on partial experiments using computer vision techniques. Omitting data augmentation, the first experiments for the breast dataset at 200X magnification to train a VGG19 network, the accuracy achieves 76%. We can conclude that data magnification and normalization are transcendental for the image classification, but sometimes we can not solve the dataset deficiencies with these techniques. For example, image normalization like Stain normalization, is specifically designed for transferring color range between histopathological images. Usually, for common problems is used a standard normalization of all the images and not using specific color ranges. In the case of data augmentation, a very popular process before training any model, the technique could not have an inverse effect in the accuracy if is applied to balance the training classes, in the worst case it will not have any effect on the final performance. For example, in our case, data augmentation probably does not have any effect to improve the SIFT feature, since the main objective of this handcrafted feature is to detect key points does not mattering rotations and orientations, data augmentation could not represent any possible improvement since the key points will be always the same.

We have discussed some of the most important strategies to tackle a problem like histopathological classification. We need to be clear at the moment of define our objective, what will be the metrics to compare, what kind of data will be analyzed, characteristics and deficiencies to be considered before training and finally the input format and process line of the training technique; image channels, calculations and expected outputs.

Table 5.1: General performance of BreakHis Dataset using all the magnifications. DP Deep Learning, TL Transfer Learning, FT Fine Tuning, CV Computer Vision. *AUC obtained after voting technique.

BreakHis			
Technique	**Features**	**Accuracy**	**AUC**
DL	**CNN**	86.47 ± 2.64	100 ± 0 *
	CNN+RNN	**90.25 ± 1.97**	**96.25 ± 2.87**
TL	**VGG19**	77.47 ± 4.66	82.6 ± 6.69
	ResNet152	68.62 ± 4.16	74.37 ± 5.63
	Incepion V3	56.04 ± 6.24	51.02 ± 4.44
	VGG + Res	77.24 ± 5.05	82.5 ± 7.01
	VGG + Res + IV3	76.69 ± 5.54	82.25 ± 6.86
FT	**VGG19**	71.41 ± 3.91	73.2 ± 9
	ResNet152	67.96 ± 6.48	70.27 ± 10.17
	Incepion V3	53.88 ± 6.41	50.52 ± 4.06
	VGG + Res	73.33 ± 6.8	75.1 ± 10.33
	VGG + Res + IV3	71.7 ± 7.09	71.9 ± 12.25
CV	**HOG**	56.46 ± 6.1	55.22 ± 6.37
	SIFT	60.97 ± 6.51	59.6 ± 7.8
	Haralick	62.26 ± 6.65	64.55 ± 7.85
	LBP	71.96 ± 10.72	79.92 ± 7.53
	LPQ	72.17 ± 8	76.75 ± 9.57
	LBP + LPQ	75.85 ± 8.72	80.82 ± 11.09
	LBP + LPQ + Haralick	75.08 ± 8.61	78.9 ± 11.96

Table 5.2: General performance of Lung Dataset using all the magnifications. DP Deep Learning, TL Transfer Learning, CV Computer Vision.

Lung			
Technique	**Features**	**Accuracy**	**AUC**
DL	**CNN**	94.8 ± 0	98 ± 0
	CNN+RNN	**99.77 ± 0**	**100 ± 0**
TL	**VGG19**	93 ± 4.65	97.2 ± 4.34
CV	**HOG**	73.2 ± 13.24	85.8 ± 10.88
	SIFT	28.04 ± 2.75	40.5 ± 3.2
	Haralick	81.82 ± 10.41	94.3 ± 4.21
	LBP	89.3 ± 8.5	96.3 ± 4.24

Table 5.3: General performance of Colon Dataset. DP Deep Learning, TL Transfer Learning, CV Computer Vision.

Colon			
Technique	**Features**	**Accuracy**	**AUC**
DL	**CNN**	**100 ± 0**	**100 ± 0**
	CNN+RNN	**100 ± 0**	**100 ± 0**
TL	**VGG19**	97.79 ± 2.21	99.1 ± 2.23
CV	**HOG**	67.27 ± 7.12	74.1 ± 7.5
	SIFT	51.44 ± 1.65	51.52 ± 2.37
	Haralick	73.54 ± 7.9	79.8 ± 8.1
	LBP	9.83 ± 7.27	96.7 ± 5.57

5.2 General Comparison

Almost 800 techniques were implemented for this project, including several configurations and combinations of deep learning, machine learning and computer vision. It could be hard trying to make a general comparison using the full ranking of all the techniques, for that reason we summarize all the outcomes according to their corresponding area, deep learning, transfer learning, fine tuning and computer vision.

The main comparison or at least the more equal according to its conditions, is the features comparison. Feature extractors like transfer learning, fine tuning and computer vision models were used to train 10 machine learning techniques. From the breast cancer, on table 5.1, which summarize all the different magnifications, we can observe that accuracy and AUC metrics are similar between each other but is quite interesting that in transfer learning and computer vision the accuracy variation increases. For transfer learning and fine tuning could be an obvious pattern since the number of features is more 250 features and these increases according to the combination of different network features in each case. For computer vision features we can observe that all the features have a big variation between each machine learning classifier, similar than fine tuning and worse than transfer learning. For example, histogram of oriented gradients, which is the handcrafted technique with more features, more than 2,000 per image, has less accuracy variation between machine learning techniques than local binary pattern, which is the handcrafted feature with less 30 features per images and the technique with the highest accuracy and also the highest standard deviation. This fact could be a consequence of different factors, for example the quality and complexity of the features or the scope of the machine learning techniques.

Considering the quality of features obtained by computer vision techniques we can try to improve the feature generation with specific parameters for each magnification factor, for example for the histogram of oriented gradients we can use a bigger matrix to analyze the images with higher magnification factor like 400X. Haralick features could be obtained using specific 2D and 3d dimension matrices per direction as has been tested on important works related with the standardization for image biomarkers. For the second possible cause, which is the scope of machine learning techniques over high dimension inputs, the solution could be

feature selection to reduce the hyperplanes and considering just the features with the highest coefficients. We can implement both solutions to reduce the outcomes dispersion and then increase the reliability of computer vision techniques.

General performance of lung a colon cancer, on tables 5.3 and 5.2, is quite similar between best techniques of computer vision and deep learning. The dispersion of the outcomes is similar at least in the area under the curve. Similar than breast features, high dispersion is present on techniques with a high feature dimensions like histogram of oriented gradients. Analyzing both datasets we can find similar patterns on these performances, but probably high dispersion is just a consequence of the bad behavior of features for object detection and segmentation. There are other possible explanations for breast performance like that some magnification factors like the 400X could demerit the quality features of magnifications like 40X and 100X which have better accuracy and area under the curve. We can then asseverate that high magnification factors are not suitable for handcrafted features and they play an important role on the feature quality.

5.3 Deep Learning and Computer Vision: Why they matter

In our project, the main argument to verify and compare hundreds of possible solutions for image classification, was the deep learning downside, hardware limitations. We mentioned that during the last 10 years, several architectures of convolutional neural networks have been released. The number of trainable parameters has been reduced or at least these parameters are the same as the first models if we compare for example AlexNet (2012) and ResNet(2015). Accuracy in complex models like ImageNet has increased year by year, following the same example, AlexNet and ResNet152, both have about 60M parameters but there is a10% difference in accuracy between each other. Training ResNet model requires 10 times more computations than AlexNet. Basically, training a deeper model requires more time and more hardware resources. Some of the improvements on current models is the use of smaller filters size, parallel convolutions, and convolutions 1x1 which have allowed models like Inception V3 to achieve a great accuracy and deepness using 5 times more computations than AlexNet. There are other networks like EfficientNet, MobileNet and CaffeNet trying to standardize different configurations in the number of layers to tackle specific problems according to the image dimension and complexity of the task, but this still being a distribution problem, using networks with less computations implies sacrificing accuracy.

Until now, we only have mentioned deep learning models, but what about hardware improvements. We could say that graphic processors improve its performance year by year and they are more accessible nowadays, then this is the best opportunity to apply deep learning. Having access to computational power does not mean necessarily have to acquire a cluster or something similar since cloud could be the solution. Several companies like Google, Amazon and many others have released projects like Colab or AWS to use hardware services remotely, some of them totally free or with prices adapted for the time and resources that you need. In our case this project was fully performed using Colab without any inconvenient.

Deep learning aspires to be the technique more used for image task in the future, but it does not imply that some other problems will disappear. Solving hardware requirements does not mean that deep learning is the best option for this kind of problems. If we use our project as main comparison, we can say that there is not any possibility to apply computer vision techniques instead of deep learning having access to hardware requirements. In complex problems like breast cancer classification, our best technique was convolutional neural networks and recurrent neural networks. For that implementation we select a deep architecture like Xception network over other architectures like inception, because it has the best proportion deepness/trainable parameters than any others. Similar than the previous topic, we reduce the image to a size of 299x299 instead of using patching technique, and we only use only channel image for recurrent networks. The main difference using other architecture, patching and 3 channel inputs could be a big disaster. If the compare the inference time even using graphic processors, the number of images generated by pathing techniques makes impossible to train a model of these characteristics, having the hardware resources does not mean that everything will be an easy and quickly implementation, even whit these resources is necessary to implement a good strategy. For problems different from pathology classification, image segmentation, detection, color normalization and data augmentation are essential factors before training any model, even for a deep learning technique.

Finally, no one can demerit deep learning potential, but there are downsides that are not solved by hardware requirements. Probably the main consideration is that these networks are a black box, in which there is not any reliability of what is considered by the network at the decision moment. Here is when external factors like water marks, similarities and amazing coincidences between class inputs could achieve a great performance for that data distribution, but, when the model is tested for inputs coming from a different distribution the outputs are a big disaster. For example, a model trained to classify full body images of men and women achieves 100% of accuracy, but when the model is trained with another dataset, it achieves something close to 50%. We could explain this like overfitting or something else, but when the human factor analyzes the input images of the first dataset, he detects that men wear white shoes and women black shoes and in the second dataset there is a high diversity of color shoes. The model detects an easy way to distinguish men from women considering an imperceptible factor like color shoes and ignoring some other factors like human complexity, hair, and facial characteristics. These problems happen more than we can imagine, and in some cases that cannot be avoided by the any possible preprocessing technique.

If some of these conditions occurs on computer vision techniques, there could be an insignificant impact, for example for histogram of oriented gradients, the main characteristic obtained could be just the shape of these shoes, for texture feature like Haralick o local binary patterns could just represent a concurrent value, similar than the concurrent value of hair and complexity. This is the main advantage of handcrafted features, that they are not trainable models, and their calculations are always the same for whole image and all the images, there is not any inference over specific characteristics of each problem image.

We can define deep learning as a double-edged sword, great potential that could not be

able to generalize, and we can define computer vision technique as a less powerful but more reliable technique. If your intention is not to generalize a model capable to solve only the input distribution of your interest, then deep learning is your best option, since does not matter what it is detecting, what really matters is the final classification. If your intention is to generalize a model capable to solve any possible input from different distributions, then deep learning could be not the best option. Of course, we cannot discard deep learning as a possible option, but the most reliable solution to achieve a generalization is increase the input data. In our case, obtain this kind of medical data is to hard and not to easy to solve like hardware requirements. For example, Campanella et al. [3] obtained almost 50,000 histopathological slides from many countries around the world to train a very deep and robust model, when this model was tested with the images from other distribution in this case the CAMELYON16, a cancer detection challenge, the model reduces its accuracy almost 8%. It could sound like there is not any generalization effect on that project, but we need to admire the great effort to recollect all the data and perform this experiment. Of course, there is a high cost of collecting all the data and training a model with all the obtained information just to get a model that decreases its accuracy 8% when is tested with different inputs. Authors make another comparison, this time using the original model, winner of that challenge trained with almost 2,000 slides, to be tested with their own test set obtaining an accuracy close to 70%, representing a drop of 20% in accuracy.

There is not an experiment of similar conditions to verify the handcrafted features performance over different image distributions, but there is an important project by Zwanenburg et al. [50], an initiative to standardize features like Haralick in medical images. Comparing the possibilities of a general model of both techniques, the best opportunity to generalize convolutional networks is obtaining all the possible data, something out of range of computational science, but what could the best way to generalize a handcrafted feature, of course the data plays an important role, but not to important like in deep learning since we have to remember that there is not a training model to obtain the features. Then a possible solution is the feature standardization proposed by Zwanenburg. Adapt these features according to a specific input is something that computational science can achieve. Predefined covariance matrices and orientations are some of the proposals done by Zwanenburg, that combined with feature selection on techniques like support vector machines could improve the outcomes. Probably the main method proposed by this initiative is the intensity metrics calculation, a method related for color diversity over this kind of images. If we remember we need to apply Stain normalization to avoid the color diversity of these images, this method could be more effective, and then reduce the inconvenience of color and structure variation. Summarizing, all the possibilities of generalize a model capable to classify any possible histopathological image of determined cancer type are focused on computer since aspects in case of handcrafted features and more related for data diversity in case of deep learning. In nutshells, deep learning has better capabilities to obtaining better outcomes than handcrafted features but less reliability than computer vision techniques. If the main intention of the problem is to classify and specific image distribution, deep learning is the best option, if the main purpose of the model is to achieve a good global performance over any possible distribution, computer science has more relevance in handcrafted features than in deep learning.

Chapter 6

Conclusion

One of the most interesting research questions that we presented at the beginning of the project was if handcrafted features could replicate the outcomes obtained by deep learning. These could be the pillar of the project since we have been looking for an alternative of deep learning, or at least a possible configuration that takes less resources and time. We confirm that fine tuning and transfer learning are not a real alternative since to obtain good results was necessary train at least more than 60% of the network and 80% to obtain similar results. In our case that were not better than the convolution network used as baseline, training 15% of different architectures the outcomes was very disappointing.

Fortunately, in the case of handcrafted features we obtain more promising results. One of the most interesting conclusions is that features related to patterns and textures like Haralick, local binary patterns and local phase quantization obtain the better performance. Even when features like SIFT or histogram of oriented gradients are more popular for the use of classification in objects. The main advantage of these features is that are easy obtained, since each image is just read it as a numerical array, then, having more or less images does has any influence on the quality of features. The fact that the process was quite simple does not meter that cannot being improved, Stain normalization demonstrate that is a great prepossessing technique, since that is probably the main difference with other publication like the author of BreakHis dataset. In our experiment we use similar features, obtaining better results with Stain normalization. Then, we can answer that hematoxylin and eosin color normalization has

Table 6.1: State of the art comparison of BreakHis Dataset .

BreakHis	**Accuracy**				**AUC**			
	40X	**100X**	**200X**	**400X**	**40X**	**100X**	**200X**	**400X**
Spanhol et al. (2017)	85.5	83.5	83.6	80.8	-	-	-	-
Spanhol et al. (2017)	83.8	82.1	85.1	82.3	84.8	81.14	86.1	84.41
Dimitropoulos et al. (2017)	91.8	92.1	91.4	90.2	-	-	-	-
Han et al. (2017)	85.6	83.5	83.1	80.8	-	-	-	-
Benhammou (2018)	89.1	89.8	87.2	85.3	-	-	-	-
Han et al. (2018)	**95.8**	**96.9**	**96.7**	**94.9**	**98.91**	**98.8**	**98.68**	**98.71**
Our Approach	**92.08**	**91.84**	**88.71**	**88.4**	**98**	**98**	**97**	**92**

big influence over classification results.

In the internal comparison, handcrafted outcomes were better than the baseline of the convolution network model. In that case, were used a very deep network, using a high increased dataset by the patching technique, doing a robust model with a patch voting process of several images ranked by the deep learning model. These process takes several time and considerable resources. We can say with an experimentation support that handcrafted features are a real alternative and achieve great outcomes in digital pathology without deep learning techniques.

Finally, the best performance was obtained by the combination of deep learning techniques, convolution and recurrent networks, these model was quite close to the state of the art as we can observe on table 6.1. That combination can be defined as a good and efficient implementation of deep learning. Designed to not being overfitted and not use more than the necessary resources at the first stage. The model takes the best of each technique, omitting the not successive techniques like patching, and adding and extra technique for the implementation. The model used considerable resources, but are used in the correct way. If we evaluate the accuracy and area under the curve we can say that they satisfied the best possible outcomes, since we were just by 2% of difference in accuracy from the state of the state of the art for 40X and 100X magnification.

We observed that support vector machine and linear support vector machines were the best machine learning techniques in the final ranking. In the case of datasets, for BreakHis we can say that the 40X and 100x magnification obtained the better accuracy's and area under the curve, and we can conclude that high magnifications are not always the best option since we could lose environment characteristics.

As general consideration of the project, we need to remember that we are the smart, intelligent and aware factor here. We cannot trust in a complex model even when this is recognized as "Deep learning" technique. That model is not more than a mathematical optimization to converge in an optimal solution. There is noting that the model can consider from the problem and values, and we need to be sure that the values processed are the correct values. Convolution networks are still being black boxes, improved models, but still being black boxes. We cannot confirm or asseverate that each weight from a filter was trained to detect cells, glands or patterns since is hard to the human eye and brain confirm all the possible information in just one image. Several times there are external factors, watermarks or elements not considered that are used by the model to train. At the end there is nothing relevant obtained by the framework than the information not desired by the user. We need to use more concrete techniques like handcrafted features, of which we know clearly what is calculated and what is obtained from each image or input. We need to use all the possible ways and techniques that can give us the certainty of what is calculated before implement a black box technique like convolutional networks. It is necessary to focus on solvers and not in solutions, understand the problem and design a plan, not implement brute force among several layers.

6.1 Future Work

There are several handcrafted features designed to tackle different problems. We consider that image processing has not taken advantage of digital pathology to demonstrate his potential. In the future, could be possible present a more complete work using other handcrafted features and architectures. With more opportunities of hardware access, a whole slide image experiment can be performance and of course evaluate the models with other cancer datasets. We really trust in the potential of these techniques, not only for digital pathology. These have been used for spoofing, human and object recognition. It could be viable that have seen the outcomes of the combined features, in the future, create an original handcrafted model using the better characteristics of the features that we implemented on this project for digital pathology recognition. Meanwhile, the medical imaging community has come together to globalize and standardise the use of these features for image classification.

6.1.1 Feature Selection for Machine Learning

Machine learning techniques were used as training models for deep learning and computer vision techniques. We implemented 10 strategies as training models for feature classification, all of them with different characteristics and level complexity, from the simplest techniques like KNN and Decision Trees to techniques implying boosting algorithms and kernels like Adaboost and Support Vector Machines. There are important factors between each of these techniques like feature selection. The main function of Support Vector Machines is to find the best hyperplane to classify elements of different classes. The dimension of these hyperplanes is the same as the number of input features minus one. Basically, each side of this plane separates each class. Closest inputs to the plane are the Support Vectors.

Vector and input element separation are the workflow of KNN and SVM. Something interesting that we need to remember is that these techniques are high contrasted, SVM obtained the best performance over all the machine learning techniques and KNN the worst performance. Both techniques share the same workflow sin similar ways, and both can be improved. Once calculated the hyperplane, it is possible to use the coefficients of the model. These coefficients are the orthogonal coordinates to the hyperplane. Direction of these coordinates represents the predicted class. If we compare the size of these coefficients to each other is possible to identify the main features used for classification. Considering that we have used several combinations of features and different output sizes in transfer learning and fine tuning and specially in histogram of oriented gradients for computer vision techniques, this coefficient comparison could improve several outcomes in these experiments where the features did not represent a real alternative for image classification.

6.1.2 Image Features for Medical Images

Different from other problems, medical images are used on different tasks, like classification, segmentation, prediction, and annotation. These activities usually require specific calculations

that are hard to specific or at least to obtain using deep learning in a direct way. For example, in our case, we apply 5 computer vision techniques with different functions and complexity levels. For example, features for texture detection are just the implementation of the correlation matrix around or a binary sampling of specific pixels, different from the object detection and segmentation features which include Gaussian and Fourier transformations, pixel magnification and orientations. If we compare these features with the rest of possible features that we have not applied like ORB (Oriented Fast and rotated Brief) and SURF (Speeded-Up Robust Features), both features for object and key points detection, we could say the feature selection was actuated and that could be something difficult trying to find other feature capable to perform the outcomes done by texture features.

More than 50 researchers around the world are currently working to define the potential of computer vision features over specific medical images. Image biomarker standardisation initiative, a project leaded by Zwanenburg et al. [50], is a great international collaboration looking for the development of features capable to perform the prediction, classification, and segmentation. Zwanenbunrg's project presents 11 different features inspired in features like Haralick textures and Local Binary Patterns, the same features that we have implemented to obtain the best accuracy using computer vision techniques, to classify non-small-cell lung carcinoma images.

Intensity-based coefficient of variation

Haralick textures calculate the covariance factor of the pixels contained around specific image sectors, according to the matrix division of the image. Zwanenburg present a feature based on this idea, including extra factors like intensity. The author uses different intensity ranges based on the mean and variance intensity calculation. Consensus comes from strong to very strong at different configuration levels of phantom matrixes and configurations like minimum intensity, maximum intensity, 10 and 9 intensity percentiles. Specifically, the intensity-based coefficient variance dividing the variance over the mean intensity distribution. Basically, the main difference between this implementation and Haralick features is the Voxel intensities within a defined neighborhood around a center voxel, this difference could be highly significant in the performance for BreakHis dataset since we have to remember that we need to apply Stain normalization to normalize the range of intensity after the gray scale transformation. This feature is probably an alternative for Stain normalization or maybe a complement of this process that could improve the performance even on datasets with high diversity of eosin and hematoxylin coloration.

Grey level co-occurrence based features

Intensity based features are not the only based on the Haralick textures or GLCM (Grey Level Co-Occurrence Matrix) features. Zwanenburg presents a variation of these features to improve rotational invariance. Authors propose five additional methods to arrive a single feature, including the original these methods are the following.

- Compute features by a 2D directional matrix and average over 2D directions and slices.
- Compute features by a single matrix after merging 2D directional matrices per slice, and then average over slices.
- Compute features by a single matrix after merging 2D directional matrices per direction, and then average over directions.
- Compute features by from a single matrix after merging all 2D directional matrices.
- Compute features by a 3D directional matrix and average over the 3D directions.
- Compute features by from a single matrix after merging all 3D directional matrices.

These steps are documented by a compendium of their corresponding neighboring grey level dependence matrix using Chebyshev distance 1 and coarseness 0. One of the principal differences on these methods are the weighted distance by multiplying the value matrix with a weighting factor. Similar than the Intensity-based coefficient of variation, there are several calculations than complement these features according to the selected process like correlation, inverse variance, dissimilarity and contrast.

There is a great diversity of configurations to apply on this problem and the expected performance is better than the original handcrafted features. There are many features focused on diverse problems, as we have mention before some of them like SURF, BRIEF, FAST and ORB are features based on SIFT features for the detection of key points at different conditions. Usually these adaptations are based on the implementation of a new technique like convolution filters, blurring and machine learning. In this case, there is something different from a new version or a new feature. Several adaptations and configurations are presented to obtain different metrics and then improve the evaluation of the feature. These configurations, like the specific grey level matrix used for each variation of the method, have two real impacts of the original feature performance, the first of course is the range of possibilities to improve the current results, but the second impact is the reduction of problem diversity that the original feature can tackle. In nutshells, our project supports the theory that texture features perform a great classification of histopathological images. Considering these aspects, future work of this project could be extended over different ways, even more extensive than the original version. Trying with other features, developing a new feature, modifying a current feature or participate on the standardization of texture features for the classification of histopathological images are some of these options. What is quite significant in the standardization of computer vision techniques for medical images is that they represent a research interest which is quite close to define a precise technique for the detection of one of the principal causes of death in the world.

Bibliography

[1] Bray, F., Ferlay, J., Soerjomataram, I., Siegel, R.L., Torre, L.A. and Jemal, A., Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. CA: A Cancer Journal for Clinicians, 68: 394-424, 2018.

[2] Chang, K., Creighton, C., Davis, C. et al. The Cancer Genome Atlas Pan-Cancer analysis project. Nat Genet 45, 1113–1120, 2013.

[3] Campanella, G., Hanna, M.G., Geneslaw, L. et al. Clinical-grade computational pathology using weakly supervised deep learning on whole slide images. Nat Med 25, 1301–1309, 2019.

[4] Nanni, L., Ghidoni, S., Brahnam, S. Handcrafted vs. non-handcrafted features for computer vision classification. Pattern Recognition, 71, 158–172, 2017.

[5] D. Bardou, K. Zhang and S. M. Ahmad, "Classification of Breast Cancer Based on Histology Images Using Convolutional Neural Networks," in IEEE Access, vol. 6, pp. 24680-24693, 2018.

[6] Sharma, S., Mehra, R. Conventional Machine Learning and Deep Learning Approach for Multi-Classification of Breast Cancer Histopathology Images—a Comparative Insight. J Digit Imaging 33, 632–654, 2020.

[7] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, pp. 248-255, 2009.

[8] Borkowski, Andrew Bui, Marilyn Thomas, L. Wilson, Catherine DeLand, Lauren Mastorides, Stephen. (2019). Lung and Colon Cancer Histopathological Image Dataset (LC25000), 2019.

[9] M. D. Bloice, P. M. Roth, and A. Holzinger, "Biomedical image augmentation using Augmentor," Bioinformatics, vol. 35, no. 21, pp. 4522–4524, Nov. 2019.

[10] Spanhol, F. A., Oliveira, L. S., Petitjean, C., Heutte, L. (2016). Breast cancer histopathological image classification using Convolutional Neural Networks. International Joint Conference on Neural Networks (IJCNN), 2016.

[11] Van den Oord, Aaron, et al. "Conditional image generation with pixelcnn decoders." Advances in neural information processing systems. 2016.

[12] Liu, Yaojie, Amin Jourabloo, and Xiaoming Liu. "Learning deep models for face anti-spoofing: Binary or auxiliary supervision." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[13] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems. 2012.

[14] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 770-778, 2016.

[15] Jason Wei, Laura Tafe, Yevgeniy Linnik, Louis Vaickus, Naofumi Tomita, Saeed Hassanpour, "Pathologist-level Classification of Histologic Patterns on Resected Lung Adenocarcinoma Slides with Deep Neural Networks", Scientific Reports;9:3358, 2019.

[16] Benhammou, Yassir, Tabik, Siham, Boujemâa, Achchab, Herrera, Francisco. A first study exploring the performance of the state-of-the art CNN model in the problem of breast cancer. 1-6, 2018.

[17] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 2818-2826, 2016.

[18] Han, Z., Wei, B., Zheng, Y. et al. Breast Cancer Multi-classification from Histopathological Images with Structured Deep Learning Model. Sci Rep 7, 4172, 2017.

[19] C. Szegedy et al., "Going deeper with convolutions," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, pp. 1-9, 2015.

[20] Jason Wei, Laura Tafe, Yevgeniy Linnik, Louis Vaickus, Naofumi Tomita, Saeed Hassanpour, "Pathologist-level Classification of Histologic Patterns on Resected Lung Adenocarcinoma Slides with Deep Neural Networks", Scientific Reports;9:3358, 2019.

[21] S. H. Shabbeer Basha, S. Ghosh, K. Kishan Babu, S. Ram Dubey, V. Pulabaigari and S. Mukherjee, "RCCNet: An Efficient Convolutional Neural Network for Histological Routine Colon Cancer Nuclei Classification," 2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV), Singapore, pp. 1222-1227, 2018.

[22] Bayramoglu, Neslihan, and Janne Heikkilä. "Transfer learning for cell nuclei classification in histopathology images." European Conference on Computer Vision. Springer, Cham, 2016.

[23] K. Sirinukunwattana, S.E.A. Raza, Y.W Tsang, I.A. Cree, D.R.J. Snead, N.M. Rajpoot, 'Locality Sensitive Deep Learning for Detection and Classification of Nuclei in Routine Colon Cancer Histology Images,' IEEE Transactions on Medical Imaging, 2016.

[24] Shin, Hoo-Chang, et al. "Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning." IEEE transactions on medical imaging 35.5, 1285-1298, 2016.

[25] Simonyan, Karen , Zisserman, Andrew. Very Deep Convolutional Networks for Large-Scale Image Recognition. 2014.

[26] Maaten, Laurens van der, and Geoffrey Hinton. "Visualizing data using t-SNE." Journal of machine learning research 9.2579-2605, Nov 2008.

[27] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, pp. 2818-2826, 2016.

[29] N. Tajbakhsh et al., "Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning?," in IEEE Transactions on Medical Imaging, vol. 35, no. 5, pp. 1299-1312, May 2016.

[30] A. Kumar, J. Kim, D. Lyndon, M. Fulham and D. Feng, "An Ensemble of Fine-Tuned Convolutional Neural Networks for Medical Image Classification," in IEEE Journal of Biomedical and Health Informatics, vol. 21, no. 1, pp. 31-40, Jan. 2017.

[31] Qu, Jia, Hiruta, Nobuyuki, Terai, Kensuke, Nosato, Hirokazu, Murakawa, Masahiro, Sakanashi, Hidenori. (2018). Gastric Pathology Image Classification Using Stepwise Fine-Tuning for Deep Neural Networks. Journal of Healthcare Engineering. 2018.

[32] Chollet, Francois. Xception: Deep Learning with Depthwise Separable Convolutions. 1800-1807, 2017.

[33] Hochreiter, Sepp, Schmidhuber, Jürgen. Long Short-term Memory. Neural computation. 9. 1735-80, 1997.

[34] Budak, Umit, Comert, Zafer, Najat, Zryan, Sengur, Abdulkadir, ÇIBUK, Musa. Computer-aided diagnosis system combining FCN and Bi-LSTM model for efficient breast cancer detection from histopathological images. Applied Soft Computing, 2019.

[35] Zainudin Z., Shamsuddin S.M., Hasan S. Convolutional Neural Network Long Short-Term Memory (CNN+LSTM) for Histopathology Cancer Image Classification. In: Agarwal S., Verma S., Agrawal D. (eds) Machine Intelligence and Signal Processing. MISP 2019. Advances in Intelligent Systems and Computing, vol 1085, 2020.

[36] Dubey, Kavita, Agarwal, Anant, Lathe, Astitwa, Kumar, Dr. Ranjeet, Srivastava, Vishal. Self-attention based BiLSTM-CNN classifier for the prediction of ischemic and non-ischemic cardiomyopathy, 2019.

[37] R. M. Haralick, K. Shanmugam and I. Dinstein, "Textural Features for Image Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, Pages 610-621, November 1973.

[38] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. 2006. Face Description with Local Binary Patterns: Application to Face Recognition. IEEE Trans. Pattern Anal. Mach. Intell. 28, Pages 2037–2041, 12 December 2006.

[39] T. Ahonen, E. Rahtu, V. Ojansivu and J. Heikkila, "Recognition of blurred faces using Local Phase Quantization," 2008 19th International Conference on Pattern Recognition, Tampa, FL, 2008, Pages 1-4, 2008.

[40] D. G. Lowe, "Object recognition from local scale-invariant features," Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, Pages 1150-1157 vol.2, 1999.

[41] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, Pages 886-893 vol. 1, 2005.

[42] A. M. Khan, N. Rajpoot, D. Treanor and D. Magee, "A Nonlinear Mapping Approach to Stain Normalization in Digital Histopathology Images Using Image-Specific Color Deconvolution," in IEEE Transactions on Biomedical Engineering, vol. 61, no. 6, Pages 1729-1738, June 2014.

[43] 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.

[44] Li, W., Manivannan, S., Akbar, S., Zhang, J., Trucco, E., McKenna, S. J. Gland segmentation in colon histology images using hand-crafted features and convolutional neural networks. 2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI), 2016.

[45] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, pp. 886-893 vol. 1, 2005.

[46] D. G. Lowe, "Object recognition from local scale-invariant features," Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, pp. 1150-1157 vol.2, 1999.

[47] R. M. Haralick, K. Shanmugam and I. Dinstein, "Textural Features for Image Classification," in IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, pp. 610-621, Nov. 1973.

[48] Ojala, T., Pietikäinen, M., Harwood, D. A comparative study of texture measures with classification based on featured distributions. Pattern Recognition, 29(1), 51–59, 1996.

[49] Ojansivu, Ville, and Janne Heikkilä. "Blur insensitive texture classification using local phase quantization." International conference on image and signal processing. Springer, Berlin, Heidelberg, 2008.

[50] Zwanenburg A, Leger S, Vallieres M, Lock S. Image biomarker standardisation initiative. arXiv preprint arXiv:1612.07003, 2019