Revista de Informática Teórica e Aplicada - RITA - ISSN 2175-2745
Vol. 29, Num. 01 (2022) 102-114

RESEARCH ARTICLE

Efficient Breast Cancer Classification Using Histopathological Images and a Simpler VGG

Classificação Eficiente do Câncer de Mama Usando Imagens Histopatológicas e uma VGG mais Simples

Marcelo Luis Rodrigues Filho[1], Omar Andres Carmona Cortes[2]

Abstract: Breast cancer is the second most deadly disease worldwide. This severe condition led to 627,000 people dying in 2018. Thus, early detection is critical for improving the patients' lifetime or even curing them. In this context, we can appeal to Medicine 4.0, which exploits machine learning capabilities to obtain a faster and more efficient diagnosis. Therefore, this work aims to apply a simpler convolutional neural network, called VGG-7, for classifying breast cancer in histopathological images. Results have shown that VGG-7 overcomes the performance of VGG-16 and VGG-19, showing an accuracy of 98%, a precision of 99%, a recall of 98%, and an F1 score of 98%.
Keywords: breast cancer — machine learning — histopathological images — convolutional neural network

Resumo: O câncer de mama é a segunda doença mais mortal do mundo. Essa condição grave resultou na morte de 627.000 pessoas em 2018. Dessa forma, a detecção precoce da doença é fundamental para melhorar a vida dos pacientes ou até mesmo curá-los. Nesse contexto, podemos recorrer ao Medicine 4.0 que explora as capacidades de aprendizado de máquina para obter um diagnóstico mais rápido e eficiente. Portanto, este trabalho tem como objetivo aplicar uma rede neural convolucional mais simples, denominada VGG-7, para classificação do câncer de mama em imagens histopatológicas. Os resultados mostraram que o VGG-7 supera o desempenho do VGG-16 e do VGG-19, apresentando uma acurácia de 98 %, uma precisão de 99 %, um recall de 98 % e um escore F1 de 98 %.
Palavras-Chave: câncer de mama — aprendizado de máquina — imagens histopatológicas — redes neurais convolucionais

[1] *Curso de Sistemas de Informação, Instituto Federal do Maranhão (IFMA), São Luis - Maranhão, Brasil*
[2] *Departamento de Computação (DComp), Instituto Federal do Maranhão(IFMA), São Luis - Maranhão, Brasil*
[1]**Corresponding author**: marceloluis@acad.ifma.edu.br
[2]**Corresponding author**: omar@ifma.edu.br
DOI: http://dx.doi.org/10.22456/2175-2745.119207 • **Received:** 10/10/2021 • **Accepted:** 24/12/2021

1. Introduction

Breast cancer is a severe disease that attacks women primarily. However, about 1% of men are also affected by it [1]. In fact, breast cancer is the most common cancer among women and the second one in general [2]. The World Health Organization (WHO) estimates that this kind of cancer impacts 2.1 million women per year with 627,000 deaths, representing about 15% of all deaths caused by cancer [3]. Thus, it is critical to perform the early diagnoses to start the treatment as fast as possible, increasing the lifetime expectation and maybe getting the patient's cure.

Diagnostic demands a specific biopsy examination, in which sections of a suspect sample are placed onto glass slides to be observed under a microscope for proper analysis. Then, the pathologist examines the tissue slides at various magnification levels to view cells, glands, nucleus and detects the resemblance of these structures with normal and diseased tissue [4], identifying the morphological characteristics of the tissue, which indicates signs of malignancies to determine if a tumor is growing as a malignant one [5].

These samples of tissue images have high variability in each class (benign or malign), so methods based on spatial exploitation of image data help extract characteristics from those histopathological images. Thus, Medicine 4.0 [6] arises proposing using technology such as computer vision and artificial intelligence to help pathologists in this task. On the one hand, the pathologist is the expert who can confirm the diagnose. On the other hand, the expert is human; therefore, he is subject to physical, psychological, and visual distresses. Therefore, computational tools become essential for helping

in providing a precise and fast diagnosis.

Thus, we are engaged in investigating how convolutional neural networks (CNNs) classify cancer in histopathological images. The main problem with using CNNs is that these architectures usually involve many layers, demanding considerable time to train them, especially when using large datasets such as ImageNet. Thus, this work proposes a simpler CNN architecture that can perform efficiently in classifying histopathological images. We used the VGG-16 as a baseline because it has presented good results, then tested a smaller configuration named VGG-7 with and without transfer learning, *i.e.*, we also trained our VGG from scratch using only the BreakHis [7] dataset.

In this context, the work is divided as follows: Section 2 shows some related works; Section 3 details the BreakHis dataset, Data Augmentation techniques, Transfer Learning, an overview about VGG architectures, and our proposal (VGG-7); Section 4 describes the configuration and the images histopathological dataset used in this work. Section 4.5 describes the results of our computational experiments, performance metrics, and discussion about the results and presents the conclusions and future work.

2. Related Works

Convolutional Neural Networks (CNNs) belong to a subgroup called Deep Learning, which has played an important role in image segmentation and classification. LeCun [8] introduced the first CNN in 1998, extending the initial concept of neural networks. Indeed, CNNs represent a milestone in image-based machine learning applications.

Since LeCun's publication, intensive research has been done in the area, including comparing CNN with regular segmentation algorithms. For instance, recently, Pereira Jr. et al. [9] presented a study in which CNNs show good results even in raw images, *i.e.*, with no pre-processing or normalization.

The ability to work with images was sooner extended to medical and biomedical applications. Thus, Artificial Neural Networks (ANNs) and Deep Learning (DL) are actually the foremost machine-learning tools in several domains such as image analysis and fault diagnosis [10]. Remarkably, the use of CNNs in the field of medical research has been demonstrated in several works.

For example, Gulshan et al. [11] proposed an ensemble architecture for detecting diabetic retinopathy by using ten neural networks; then, the final results are giving by a linear average over predictions. Gayathiri et al. [12] proposed a simple CNN devised by only six convolutional layers for classifying diabetic retinopathy as well. Furthermore, Su et al. [13] improved the R-CNN to identify lung nodules in computer tomographies. These works are only some examples of many other investigations that have been conducted using deep neural networks.

Notably, in the field of breast cancer detection, [7] introduces a public dataset [1] composed of 7909 images of 82 patients, which has been the primary dataset in testing machine learning algorithms, including CNN architectures presented by the same author in [14].

In this particular field of breast cancer, Bejnordi et al. [15] evaluate some CNNs, such as, GoogLeNet, ResNet, and VGG-16, for identifying and classifying breast cancer metastasis in the context of the CAMELYON16 challenge. Silva and Cortes [16] evaluated ResNet-18, ResNet-152, and GoogLeNet for classifying breast cancer using histopathological images. Ismail and Sovuthy [17] compared ResNet-50 and VGG16 for breast cancer detection using mammograms. Furthermore, Singh et al. [18] investigated the issue of imbalanced data in datasets using VGG19.

Regarding VGGs architectures [19], a popular CNN network, it was introduced in 2015. Since then, several works deal with their classification ability or try to improve it, especially in biomedical applications. For instance, [20] compares VGG-16 and VGG-19 against ResNet50 with and without transfer learning, showing that VGGs are more efficient than the ResNet. In [21], a study on the performance of VGG16, VGG19, ResNet-50, and GoogLeNet-V3 is carried out in fine-needle aspiration cytology images, in which GoogLeNet-V3 reached the best results after a fine-tuning. Further, a novel attention-based deep learning model using VGG-16 is proposed by [22] to improve COVID-19 classification using x-ray images getting the best results. Furthermore, a modified VGG, called MVGG, is proposed and implemented in [23] to increase the detection ability on 2D and 3D mammogram image datasets.

As we can see, the field of using and studying deep learning models in the classification of biomedical images, especially the VGG architecture, is vast. Thus, in this paper, we propose a simpler VGG architecture, called VGG-7, to provide a faster training deep learning model that also overcomes the efficiency of the classical VGGs.

Next, we presents all methods used in this work, including our proposal.

3. Materials and Methods

3.1 Convolutional Neural Networks

A regular CNN is usually composed of three kinds of layers: convolutional, pooling, and fully connected. The convolutional layer produces a convolved feature matrix, also known as feature maps. The process is to apply a filter sliding the filter matrix into the input image to produce the feature map, which can be the same size as the input size if padding is using or smaller otherwise. Figure 1 shows a $5 \times 5 \times 1$ image passing through a $3 \times 3 \times 1$ filter with no passing producing a $3 \times 3 \times 1$ feature map. In colour image with 3 channels, the convolution operations extract informative features by blending cross-channel and spatial information together and each

[1] The dataset is available in ⟨http://web.inf.ufpr.br/vri/breast-cancer-database⟩

channel of a feature map is considered as a feature detector.

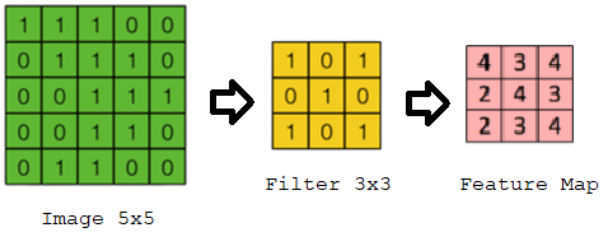


Figure 1. A 4×4 image passing through a convolutional layer

Those filters that scan over a portion of the image and extract features such as colors, shapes, and edges that ultimately define a specific image [24]. We can have as many convolutional layers as necessary.

If more than one convolutional layer is required, then we can add pooling layers between them. In essence, the pooling layer takes the feature maps produced in the convolution layer and "pools" them into an image [24], performing a dimensionality reduction. The reduction is made by using a single operation, such as the maximum (max-pooling) or the average (avg-pooling) values inside a box produced by the convolutional layer. Figure 2 shows an 5×5 image passing through a max-pooling operation reducing its dimensionality to 2×2 matrix.

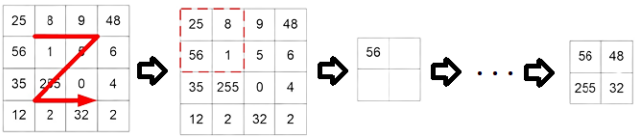


Figure 2. A 5×5 image passing through a max pooling layer

Finally, the output of the last pooling layer is flattened into a fully connected neural network. Then, a general convolutional neural network presents a shape similar to Figure 3.

3.2 Transfer Learning

Medical images are generated by special medical equipment, and their labeling often relies on experienced doctors. Therefore, in many cases, it is expensive and hard to collect sufficient training data. In such a case, transfer learning technology can be utilized for medical imaging analysis. A commonly used transfer learning approach is to pre-train a neural network on the source domain (e.g., ImageNet, an image database containing more than fourteen million annotated images with more than twenty thousand categories) and then fine-tuning it based on the instances from the target domain.

Torrey & Shavlik [25] defines transfer learning as the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned. However, what are the benefits of using transfer learning? We would state three main advantages:

- We can use models that were carefully designed by experts;
- Because experts created those models, we do not need to worry about what architecture or layers to use or include;
- Due to their careful design, they tend to perform well in image detection.

3.3 The VGG Architecture

There are essentially two VGG architectures: VGG16 and VGG19, which are the most famous and commonly used for image detection. The number in front of the name stands for the number of weighted layers in the network, being created to simulate the relation of depth with the representational capacity of the network.

The VGG-16 is devised by 13 convolutional layers and 3 Fully Connected layers as presented in Table 1. Further, the VGG-19 is composed of 16 convolutional layers and 3 Fully Connected layers as illustrated in Table 2.

Table 1. VGG-16 Architecture

Layer	Output Size	Parameter
conv1	150x150x64	kernel 3, stride 1, pad 0
conv2	150x150x64	kernel 3, stride 1, pad 0
maxpool1	75x75x64	-
conv3	75x75x128	kernel 3, stride 1, pad 0
conv4	75x75x128	kernel 3, stride 1, pad 0
maxpool2	37x37x128	-
conv5	37x37x128	kernel 3, stride 1, pad 0
conv6	37x37x256	kernel 3, stride 1, pad 0
conv7	37x37x256	kernel 1, stride 1, pad 0
maxpool3	18x18x256	-
conv8	18x18x512	kernel 3, stride 1, pad 0
conv9	18x18x512	kernel 3, stride 1, pad 0
conv10	18x18x512	kernel 1, stride 1, pad 0
maxpool4	9x9x512	-
conv11	9x9x512	kernel 3, stride 1, pad 0
conv12	9x9x512	kernel 3, stride 1, pad 0
conv13	9x9x512	kernel 1, stride 1, pad 0
maxpool5	4x4x512	-
*gap**	512	-
fc1	4096	4096 units
fc2	4096	4096 units
fc3	1	1 sigmoid

*gap: Global Average Pooling

Overall, VGGs are Convolution Neural Networks based on a spatial filter to explore the relation of different convolution filters with the neural network's learning based on the characteristics extracted. Therefore, adjusting filters and the

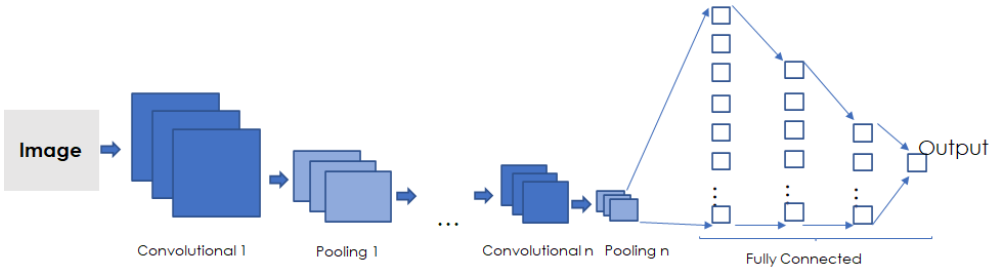


Figure 3. A general view of a CNN with *n* convolutional-pooling layers and a fully connected one

Table 2. VGG-19 Architecture

Layer	Output Size	Parameter
conv1	150x150x64	kernel 3, stride 1, pad 0
conv2	150x150x64	kernel 3, stride 1, pad 0
maxpool1	75x75x64	-
conv3	75x75x128	kernel 3, stride 1, pad 0
conv4	75x75x128	kernel 3, stride 1, pad 0
maxpool2	37x37x128	-
conv5	37x37x128	kernel 3, stride 1, pad 0
conv6	37x37x256	kernel 3, stride 1, pad 0
conv7	37x37x256	kernel 1, stride 1, pad 0
conv8	37x37x256	kernel 1, stride 1, pad 0
maxpool3	18x18x256	-
conv9	18x18x512	kernel 3, stride 1, pad 0
conv10	18x18x512	kernel 3, stride 1, pad 0
conv11	18x18x512	kernel 1, stride 1, pad 0
maxpool4	9x9x512	-
conv12	9x9x512	kernel 3, stride 1, pad 0
conv13	9x9x512	kernel 3, stride 1, pad 0
conv14	9x9x512	kernel 1, stride 1, pad 0
conv15	9x9x512	kernel 1, stride 1, pad 0
conv16	9x9x512	kernel 1, stride 1, pad 0
maxpool5	4x4x512	-
*gap**	512	-
fc1	4096	4096 units
fc2	4096	4096 units
fc3	1	1 sigmoid

*gap: Global Average Pooling

fully connected network in VGGs can perform well on different levels of information from histopathological images to deal with the variability inter-class. For example, the VGG-16 and VGG-19 stack smaller filters 3x3 to induce the effect of the large filter, but the use of small filters provided the benefits of low computational complexity by reducing the number of parameters.

On the other hand, the main limitation related to traditional VGGs was the use of millions of parameters, mainly because of computationally expensive fully connected layers, making it computationally expensive and challenging to deploy them on low-resource systems.

Moreover, VGG-16 and VGG-19 regulate network complexity by introducing 1x1 filters, which are essentially a linear combination or linear projection (the number of input and output channels is the same) and one more non-linearity onto the space of feature maps generated by previous convolutions layers. Also, max-pooling is placed after the convolutional layer, while padding was performed to maintain the spatial resolution [26]. In this case, VGG is a feature learner that can automatically extract discriminating features from histopathological images by varying the width (number of channels), depth (number of layers), and Fully Connected Layer from a convolutional neural network-based VGG architecture.

3.4 Our Proposal: VGG-7

The VGG-7 model consists of four convolution layers divided into two blocks, followed by max-pooling layers. Max-pooling can divide the images into several blocks of the same size and only take each block's higher value. Also, the global contextual information with embedded channel-wise statistics was gathered with a global average pooling [27] layer. The three last layers are Fully-Connected (FC) ones: the first two ones have 128 and 64 units, respectively, and the third one performs binary classification with a sigmoid function as presented in Figure 4.

Figure 4. VGG-7 Architecture with 4 convolutional layers and a three layered fully connected one

Our proposal uses 3x3 zero-padding convolutions layers with stride 1, each followed by a rectified linear unit (ReLU) to avoid the saturation of gradients propagation and one 2×2 max pooling operation with stride equals 2 for feature extraction. Either, we double the number of feature channels on each pooling operation step. Then, during the training process, the input to VGG-7 is fixed-size with a 150×150 RGB image.

Figure 5 shows some examples of features generated by filters from VGG-7 applied to a histopathological image of malignant breast cancer as input. Abstract and compact representations edges are extracted in the last layers. Also, the

representations downstream start highlighting what the network pays attention to, and some features are not activated, *i.e.*, most are set to zero, and it composes a sparse matrix in the feature space.

Furthermore, the width of convolutions layers (the number of channels) started from 32 in the first layer and then increased by a 2-factor after each max-pooling layer until it reaches 64. The units of the last layers are relatively small to reduce computational cost. The first Fully-Connected (FC) layer has 128 units that correspond to double the filter from the previous max-pooling layer, and the second layer has 64 units. The final layer is the Sigmoid layer responsible for the binary classification. The structure and hyperparameters of each layers are shown in Table 3.

Table 3. VGG-7 Architecture

Layer	Output Size	Parameter
conv1	150x150x32	kernel 3, stride 1, pad 0
conv2	150x150x32	kernel 3, stride 1, pad 0
maxpool1	75x75x32	size 2, stride 2
conv3	75x75x64	kernel 3, stride 1, pad 0
conv4	75x75x64	kernel 3, stride 1, pad 0
maxpool2	37x37x64	size 2, stride 2
globalAvgPool	–	–
fc1 and fc2	–	128 and 64 units
fc3	–	1 sigmoid

Next, we detail the experiments and discuss the results.

4. Computational Experiments

4.1 Dataset

The dataset comes from Breast Cancer Histopathological Dataset [7], a public domain dataset made available by the Laboratory of Vision, Robotics, and Images from Universidade Federal do Paraná (UFPR). The dataset comprises 7909 images of tumoral tissues from 82 different patients. The images have different zoom magnitudes devised by 40x, 100x, 200x, and 400x as presented in Fig. 6, which shows a malignant tumor.

Either, images are in the "png" format, having a resolution of $700 \times 460 pixels$, three RGB channels, and 8 bits depth in each one. Table 4 presents the number of benignant and malignant tumors according to the magnitude.

Table 4. Dataset Structure

Magnitude	Benign	Malign	Sub-Total
40X	652	1370	1995
100X	644	1437	2081
200X	623	1390	2013
400X	588	1232	1820
Total	2480	5429	**7909**

4.2 Data Augmentation

When the database is small and transfer learning seems insufficient to train the CNN properly, we can use a data augmentation method to improve the dataset size. Also, it is essential to teach the network the desired invariance and robustness properties when only a few training samples are available. We perform standard in-place or real-time data augmentation techniques such as random rotation, random horizontal and vertical flip, width and height shift range.

The images were transformed during the training to save memory, but this resulted in slower training. The images are converted from RGB to BGR, and then each color channel is zero-centered without scaling. The complete set of operations are: randomly flip horizontally, randomly flip vertically, random rotations of 50 degrees, width, and height shift with filling the remaining area with the nearest pixels. The data augmentations parameters are summarized in Table 5.

Table 5. Parameters of data augmentation

Parameters of Image Augmentation	Values
Rotation range	50
Width shift range	0.2
Height shift range	0.2
Horizontal flip	*True*
Vertical flip	*True*
Fill mode	*nearest*

4.3 Experiments Setup

The application was implemented in Python 3.0 using recent versions of TensorFlow[28] and Scikit-learn [29] library. The code and the training step have been done in Kaggle [30], which was essential to this work because this platform provides a GPU Nvidia Tesla P100 used for training the CNN. The virtual machine is a two CPUs Intel® Xeon 2.30 GHz, 14 GB of RAM, and 37.11 GB of HD. Even though we used GPUs, the training step takes about 1 hour for each network configuration.

The input images and their corresponding labels are used to train the network with Adam optimizer [31] and Binary Cross-Entropy loss function (Eq. 1). All the kernels are initialized with a random uniform distribution procedure of Xavier [32]. We use class weighting to create a model where loss values for classes "benignant" and "malignant" are multiplied by their corresponding weight values to avoid unbalanced classes in the dataset. In binary classification, class weights can be represented by calculating the frequency of the positive and negative classes and then inverting it so that when multiplied to the class loss, the underrepresented class has a much higher error than the majority class.

$$J(\theta) = CE(y,\hat{y}) = -\sum_{i=1}^{\mathrm{d}} y_i \log(\hat{y}_i) + (1-y)\log(1-\hat{y}) \quad (1)$$

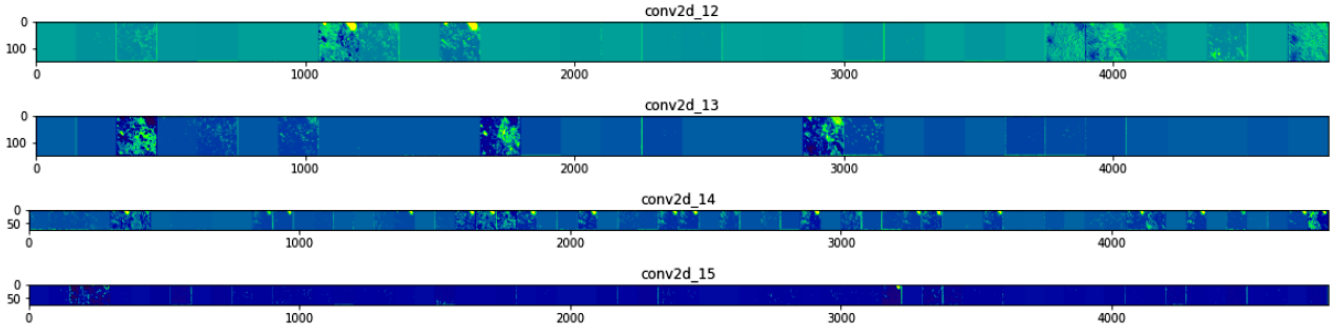


Figure 5. Example of feature extraction

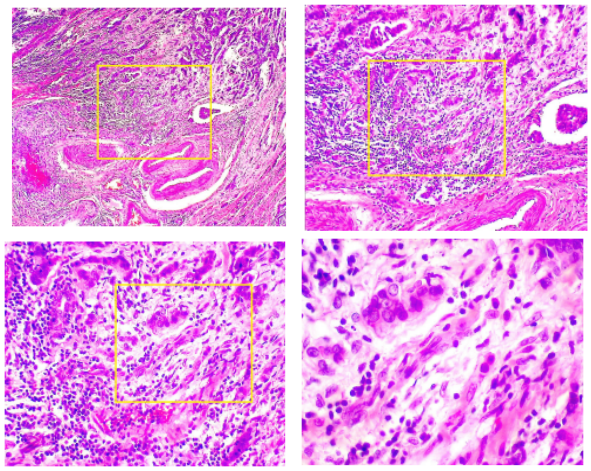


Figure 6. Malignant cancer in different magnitudes (40x, 100x, 200x, and 400x, left to right)[7]

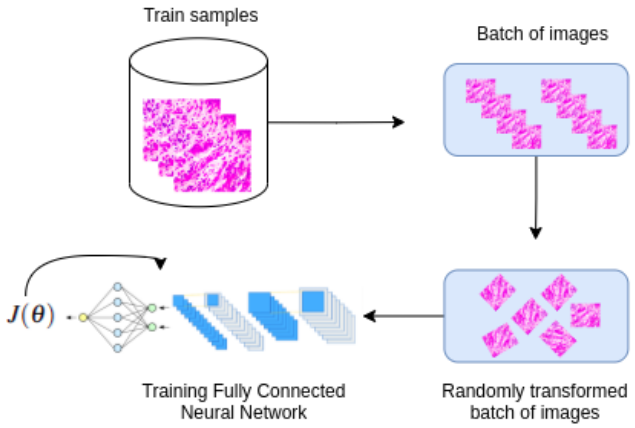


Figure 7. Training flow

The learning rate is Adam's default, 0.001, the momentum equals 0.9, and gamma equals 0.1. We train the neural network by slicing the data into batches of size 32 and repeatedly iterating over the entire dataset for 50 epochs (Figure 7). Moreover, we tested two types of sampling: hold-out (80/20 and 90/10) and k-fold ($k = 10$ and $k = 5$) cross-validation. Thus, Figure 7 presents the general procedure of training the CNNs.

Moreover, we did not use transfer learning in VGG-7 because of a lack of computing resources to run experiments with the ImageNet dataset.

4.4 Performance metric

Performance measurement is the process of collecting, analyzing, and/or reporting information regarding the performance of an individual, group, organization, system, or component [33]. The most commonly used performance metrics for classification problems in Machine Learning systems are Accuracy, Precision, Recall, and F1 score.

These metrics consider the True Positives (correctness of the predictive model for malignant tumors), False Positives (predictive model errors for malignant tumors), True Negatives (correctness of the predictive model for benignant tumors), and False Negative (predictive model errors for benignant tumors).

Accuracy

Accuracy measures the fraction of predictions our model got right. For binary classification, accuracy can also be calculated in terms of positives and negatives in Eq. 2.

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{2}$$

Precision

Precision measures what proportion of correct positive (malignant tumor) identifications is produced by the classifier, and it is expressed by Eq. 3.

$$Precision = \frac{TP}{TP+FP} \tag{3}$$

Recall

Recall or sensitivity is the actual positive rate or hit rate that measures what percentage of actual positives (cancer) is being

correctly classified (Eq. 4).

$$Recall = \frac{TP}{TP+FN} \tag{4}$$

F1 score

F1 score or F_measure is a weighted harmonic mean of precision and evoke to deal with the threshold between the two measures (precision and recall) as depicted in Eq. 5.

$$F_measure = \frac{precision \times recall}{precision + recall} \tag{5}$$

ROC curve and Area Under Curve

A ROC (Receiver Operating Characteristic) curve is a graphical plot that indicates the performance of a classification model at all classification thresholds of the False Positive Rate and True Positive Rate. True Positive Rate (TPR) is a synonym for recall and was previously defined in Eq. 4. False Positive Rate (FPR) is defined in Eq. 6.

$$FPR = \frac{FP}{FP+TN} \tag{6}$$

Given the ROC curve, it is possible to compute its area under the curve, also known as AUC, which measures the probability of a model misclassify a sample as a random positive (cancer) rather than a random negative (benignant tumor). Furthermore, AUC provides an aggregate performance measure across all possible classification thresholds, ranging from 0 (wrong predictions) to 1 (good predictions).

4.5 Results and Discussion

Hold-Out

The first result regards the losses in the training stage using only hold-out sampling. Figure 8 shows the loss as epochs increase in 80/20 (a) and 90/10 (b). As we can see, the error decreases faster and more than the other ones in VGG-7, reaching a value close to zero.

Figure 9 shows the ROC curve of the three CNN architectures with the areas of 0.94, 0.87, and 0.89 for VGG-7, VGG-19, and VGG-16, respectively in 80/20 hold-out. Then, considering that the ROC curve presents the probability of confirming the illness's presence, the figure validates the efficiency of our proposal.

In order to determine the efficiency of the CNN networks, Tables 6 and 7 present all metrics obtained by the experiment using the hold-out sampling using the rate of 80/20 and 90/10 with and without data augmentation, respectively. As we can see, the VGG-7 overcame the classical VGGs in all metrics with and without data augmentation, meaning that VGG-7 is more efficient than the other networks with fewer misclassifications. As expected, the generalization error tends to decrease as we increase the training set size in all architectures. Nonetheless, VGG-7 overcomes the other ones in all metrics.

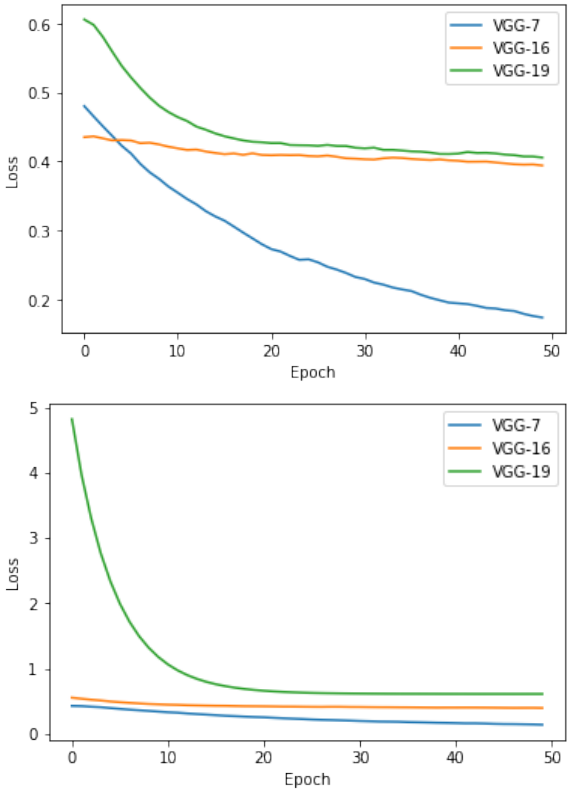


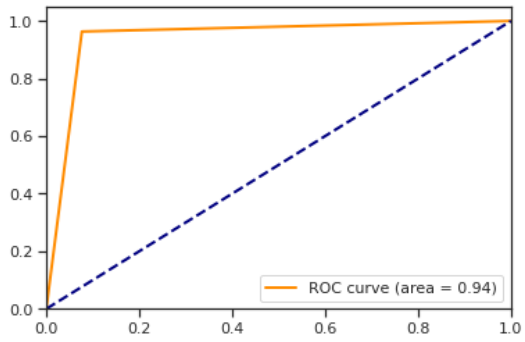
Figure 8. Training loss as the epoch count increases in hold-out 80/20 and 90/10, respectively

Table 6. Metrics - Hold-Out: 80/20 and 90/10 with no data augmentation

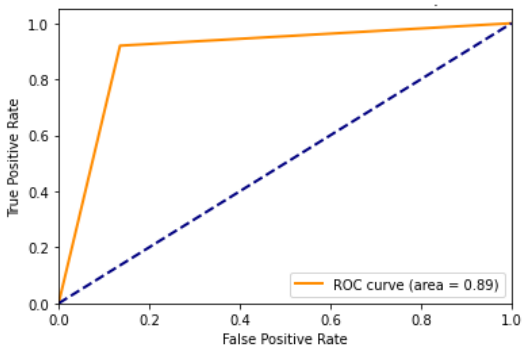
Hold-out 80/20			
Metrics	VGG-7	VGG-16	VGG-19
Accuracy	**0.95**	0.85	0.89
F1 score	**0.96**	0.88	0.92
Precision	**0.97**	0.95	0.91
Recall	**0.96**	0.82	0.93
Hold-out 90/10			
Metrics	VGG-7	VGG-16	VGG-19
Accuracy	**0.93**	0.89	0.89
F1 score	**0.95**	0.92	0.92
Precision	**0.97**	0.91	0.90
Recall	**0.93**	0.93	0.94

K-Fold Cross Validation: 10 and 5 folds

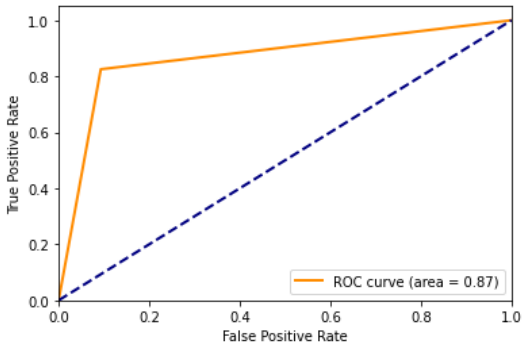
K-fold Cross-validation is a way to measure and evaluate the estimator performance. This technique randomly assigns training sets in which each training set or fold is used to train the classifiers independently. The measuring of classification metric must be computed against the test set in that fold. Finally, the result of the experiment is obtained by average them folder evaluation results. In addition, this technique requires more computing power than training in a single train



(a) VGG-7



(b) VGG-16



(c) VGG-19

Figure 9. ROC Curves

and test set such as hold-out; however, the primary benefit is that the estimators do not overfit to a single training set.

Concerning k-fold cross-validation sampling, Tables 8, 9, and 10 show the results of the 10 Stratified K-Fold Cross-Validation, *i.e.*, the stratified re-sampling guarantees that the same distribution is used in training and test sets. As we can see, the VGG-7 outperforms the other VGG version in all metrics. Unfortunately, the experiments with data augmentation and no transfer learning are blank because of our limited computing resources and user constraints. Also, we did not use transfer learning in VGG-7.

Figures 10, 11, 12 summarize the results of Area Under the ROC Curve measures in 5 Stratified K-Fold Cross Validation experiment and we conclude that the VGG-7 has a better

Table 7. Metrics - Hold-Out: 80/20 and 90/10 with data augmentation

Hold-out 80/20			
Metrics	VGG-7	VGG-16	VGG-19
Accuracy	**0.94**	0.85	0.83
F1 score	**0.96**	0.90	0.89
Precision	**0.95**	0.84	0.83
Recall	**0.97**	0.96	0.96
Hold-out 90/10			
Metrics	VGG-7	VGG-16	VGG-19
Accuracy	**0.95**	0.86	0.69
F1 score	**0.96**	0.91	0.81
Precision	**0.96**	0.86	0.69
Recall	**0.96**	0.96	1.00

ability to discern between images of the two classes.

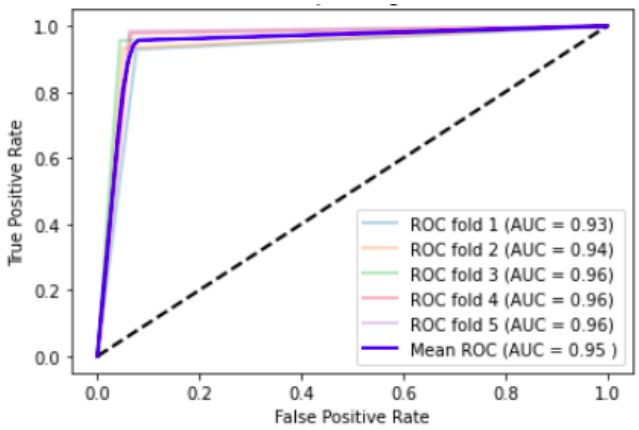


Figure 10. ROC by folds: VGG-7

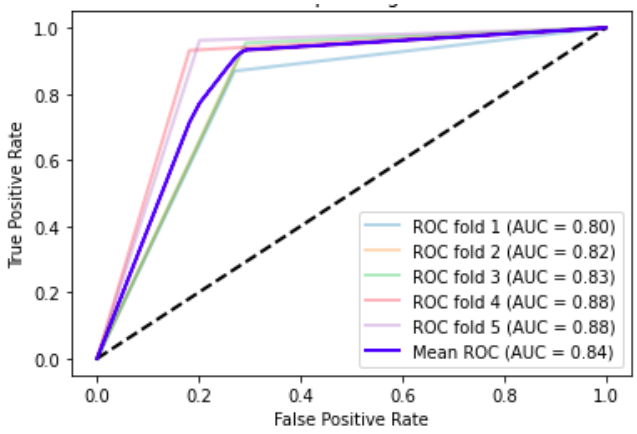


Figure 11. ROC by folds: VGG-16

Also, we perform an experiment using a $k = 5$ in order to show the performance of VGGs with no transfer learning. Tables 11, 12, and 13 show the results according the considered metrics. Moreover, the referred tables present the execution

Table 8. Results for 10 folds (k = 10): Model VGG-16

Data Augmentation			
Model		No transfer learning	Transfer learning
VGG-16	Accuracy	*	0.89 ± 0.03
	Precision	*	0.90 ± 0.02
	F1 score	*	0.92 ± 0.02
	Recall	*	0.94 ± 0.02
	AUC	*	0.85 ± 0.03
Original dataset			
Model		No transfer learning	Transfer learning
VGG-16	Accuracy	0.46 ± 0.18	0.97 ± 0.05
	Precision	0.27 ± 0.34	0.98 ± 0.02
	F1 score	0.33 ± 0.40	0.98 ± 0.03
	Recall	0.40 ± 0.49	0.98 ± 0.05
	AUC	0.50 ± 0.00	0.86 ± 0.03

Table 9. Results for 10 folds (k = 10): Model VGG-19

*the results to data augmentation experiments with no transfer learning are in blank because of our limited computing resources or restrictions.

Data Augmentation			
Model		No transfer learning	Transfer learning
VGG-19	Accuracy	*	0.88 ± 0.02
	Precision	*	0.90 ± 0.02
	F1 score	*	0.91 ± 0.02
	Recall	*	0.92 ± 0.03
	AUC	*	0.85 ± 0.03
Original dataset			
Model		No transfer learning	Transfer learning
VGG-19	Accuracy	0.50 ± 0.19	0.96 ± 0.04
	Precision	0.34 ± 0.34	0.97 ± 0.03
	F1 score	0.41 ± 0.41	0.97 ± 0.02
	Recall	0.50 ± 0.50	0.98 ± 0.02
	AUC	0.5 ± 0.0	0.98 ± 0.04

Table 10. Results for 10 folds (k = 10): Model VGG-7

VGG-7		
Metric	Data Augmentation	Original Dataset
Accuracy	$\mathbf{0.96 \pm 0.02}$	$\mathbf{0.98 \pm 0.02}$
F1 score	$\mathbf{0.97 \pm 0.02}$	$\mathbf{0.98 \pm 0.02}$
Precision	$\mathbf{0.97 \pm 0.01}$	$\mathbf{0.99 \pm 0.01}$
Recall	$\mathbf{0.97 \pm 0.03}$	$\mathbf{0.98 \pm 0.03}$

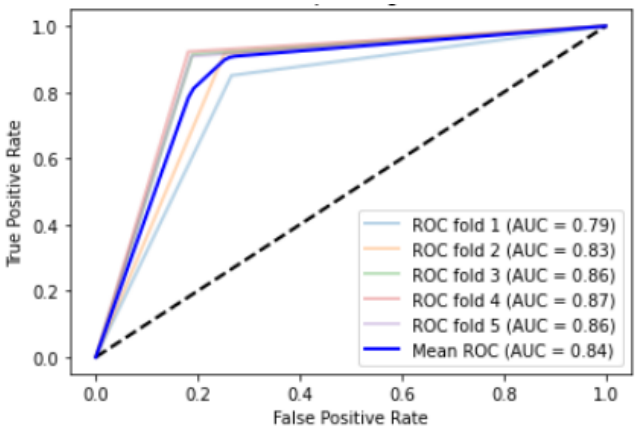


Figure 12. ROC by folds: VGG-19

time, showing that VGG-7 took much less time in the experiment, and the usage of transfer learning favors the other VGGs. On the other hand, the use of data augmentation did not improve results and increased training time complexity in quadratic order.

All in all, our proposal presents a significant advantage

Table 11. Results for 5 folds (k = 5): Model VGG-16

Data Augmentation			
Model		No transfer learning	Transfer learning
VGG-16	Accuracy	0.39 ± 0.15	0.87 ± 0.02
	Precision	0.14 ± 0.27	0.97 ± 0.02
	F1 score	0.16 ± 0.33	0.91 ± 0.01
	Recall	0.20 ± 0.40	0.94 ± 0.01
	Execution time	4h 33min 26s	3h 7min 52s
Original dataset			
Model		No transfer learning	Transfer learning
VGG-16	Accuracy	0.46 ± 0.18	0.97 ± 0.04
	Precision	0.33 ± 0.40	0.98 ± 0.03
	F1 score	0.27 ± 0.34	0.98 ± 0.03
	Recall	0.40 ± 0.49	0.98 ± 0.03
	Execution time	1h 30 min 10s	29min 16s

Table 12. Results for 5 folds (k = 5): Model VGG-19

Data Augmentation			
Model		No transfer learning	Transfer learning
VGG-19	Accuracy	0.39 ± 0.15	0.87 ± 0.03
	Precision	0.14 ± 0.27	0.90 ± 0.02
	F1 score	0.20 ± 0.4	0.90 ± 0.03
	Recall	0.14 ± 0.27	0.90 ± 0.03
	Execution time	4h 22min 35s	3h 37min 48s
Original dataset			
Model		No transfer learning	Transfer learning
VGG-19	Accuracy	0.54 ± 0.18	0.97 ± 0.05
	Precision	0.41 ± 0.34	0.98 ± 0.03
	F1 score	0.49 ± 0.40	0.98 ± 0.03
	Recall	0.60 ± 0.49	0.98 ± 0.03
	Execution time	1h 44min 16s	35min 22s

over the other VGGs because our approach needs to search for much lesser parameters than the other ones. The total number of parameters is 82,209, 15,765,313, and 21,075,009 for VGG-7, VGG-16, and VGG-19. Consequently, the training stage of VGG-7 is faster than the other ones. Additionally, our proposal requires much less memory in both the training stage and deployment or embedment.

Next, we illustrate how the VGG-7 works with activation maps using Grad-CAM.

4.6 VGG-7 visualization with Grad-CAM

Grad-CAM [34] is a method that uses a gradient to visualize and calculate the importance of spatial locations given by CNN. However, it can also be used in the Natural Language Processing domain. Thus, we are able to analyze the activation maps of the VGG-7 to different images of different classes.

As previously presented, an input image is passed through the convolutional layers and then through a pooling layer for specific computations to obtain a score for a particular category. The gradients are calculated concerning a unique class; the Grad-CAM result clearly shows attended regions. Therefore, this method is essential to note the regions that the Convolutional Neural Network considers necessary for class prediction [35].

Further, the neurons in the last convolutional layer in Convolutional Neural Networks look for high-level semantics (class-specific information) and low-level spatial features in the image, i.e, the object parts. The Grad-CAM method uses the gradient information flowing into the last convolutional layer of the Convolutional Neural Network to assign importance to each neuron for a particular class that is viewable with heatmaps.

In this context, the Grad-CAM method expressed in Equation 7 first computes the gradient of the score for specific class c(y^c) concerning the feature map activations A^k of the k̂th convolutional layer. These gradients flowing back are global

Table 13. Results for 5 folds (k = 5): Model VGG-7

Data Augmentation		
Model		No transfer learning
VGG-7	Accuracy	0.95 ± 0.02
	Precision	0.97 ± 0.01
	F1 score	0.96 ± 0.02
	Recall	0.96 ± 0.02
	Execution time	2h 54min 23s
Original dataset		
Model		No transfer learning
VGG-7	Accuracy	0.97 ± 0.03
	Precision	0.99 ± 0.02
	F1 score	0.98 ± 0.03
	Recall	0.97 ± 0.03
	Execution time	19min 48s

average pooled over spatial dimensions (width and height) to obtain the neuron importance weights α_k^c of a feature map k for a target class c.

$$\alpha_k^c = GAP(\nabla y^c(A^k)) \tag{7}$$

Finally, the Grad-CAM performed the weighted combination of activations maps in forwarding propagation with respective weight importance matrix resulting from the equation 7, followed by ReLU [36] to highlight feature with *positive* influence on the class c of interest, i.e., pixels whose intensity should be increased in order to increase y^c. Negative pixels are likely to belong to other categories in the image [34]. This process is summarized in Equation 8 that outputs a coarse heatmap of the same size as the convolutional features maps (75x75 in this case).

$$output^c = ReLU(\sum_k \alpha_k^c A^k) \tag{8}$$

Figure 13 clearly shows that the masks (Equation 8) generated by Grad-CAM and VGG-7 pays more attention to regions that cover the tissue, maybe inter-cellular regions when is predicting a malignant tumor. However, we also can see that this network did not consider all regions covering the tissue when predicting the benignant class.

5. Conclusions

This paper proposed a simpler VGG neural network called VGG-7 for classifying breast cancer in histopathological images. We trained the VGG-7 from scratch with no transfer learning because using ImageNet to train the neural network would consume too much time. Our proposal was compared against the classical VGGs (VGG16 and VGG19) with and without transfer learning and with and without data augmentation. Unfortunately, some architectural constraints hinder our experiment with data augmentation and no transfer learning in the 10-folded experiment.

All in all, classical VGGs with no transfer learning achieved lower performance than using it regardless the number of folds. Furthermore, our proposal achieved 95% of accuracy, 97% of precision, 96% of recall, and 96% of F1 Score in 80/20 hold-out sampling with no data augmentation. Further, the VGG-7 achieved 0.94% of accuracy, 95% of precision, 97% of recall, and 96% of F1 Score in 80/20 hold-out sampling with data augmentation.

Regarding the k-fold, surprisingly, the VGG-7 reached better results with the original dataset, *i.e.*, with no data augmentation, getting 97% of accuracy, 99% of precision, 97% of recall, and 98% of F1 Score, which is much better than classical VGGs with no transfer learning.

Finally, the VGG-7 presented a much better execution time in the training stage. The reached time is 19min48s in the original dataset and 2h 54min 23s using data augmentation against 3h 7min 52s and 29min 16s of VGG with transfer learning; and, against 3h 37min 48s and 35min 22s in VGG-19 also with no transfer learning. Values with no transfer learning in VGG-16 and VGG-19 are even higher.

In this context, we can summarize our contributions as:

- We proposed a simple yet effective VGG-7 convolutional neural network with fewer layers and fewer filters than other VGGs;
- We validated the effectiveness of VGG-7 by using hold-out and k-fold cross-validation with stratified folds;
- We investigated how transfer learning impact the performance of VGG-16 and VGG-19.
- We investigated the use of data augmentation in VGG-7, VGG-16, and VGG-19.

Future work includes: (i) compare the VGG-7 against other CNNs architectures; (ii) test the VGG-7 using other fully connected networks; (iii) add different classifiers, such as SVM replacing the VGG-7 fully connected layer; (iv) compare the VGG-7 using transfer learning from ImageNet and from and (v) use VGG-7 with other algorithms creating ensemble classifiers in order to improve the efficiency of the classification.

Author contributions

- Marcelo Luis Rodrigues Filho: implemented the VGG-7 architecture and made the comparison against the classical VGG-16 and VGG-19, and wrote the draft of this paper.
- Omar Andres Carmona Cortes: formulated the idea of the VGG-7, wrote, extended, and organized the final version of this work, revised the English grammar, and managed the project.

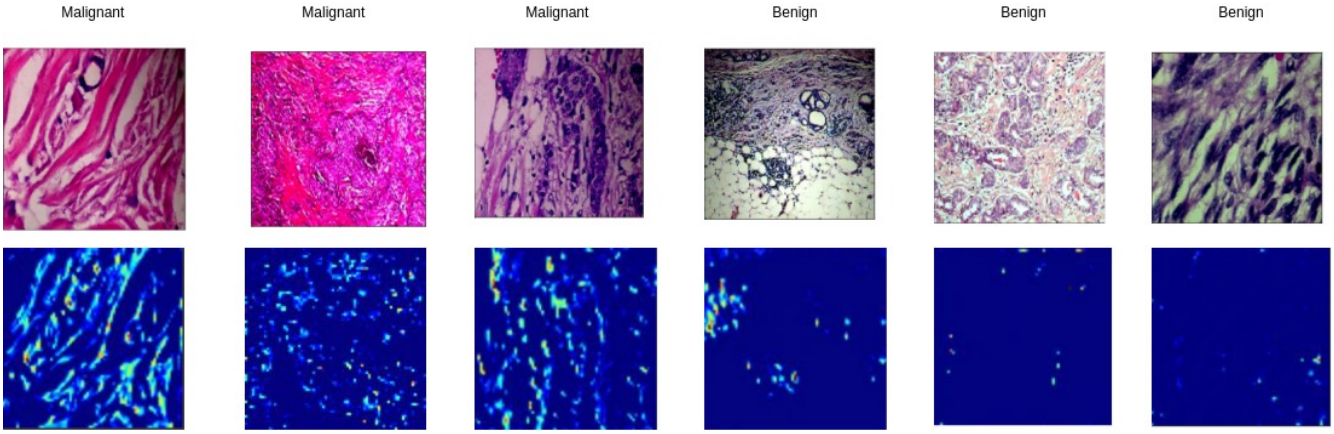


Figure 13. Grad-CAM [34] to VGG-7

References

[1] INCA. *Ministério da Saúde - Instituto Nacional de Câncer, Câncer de mama: vamos falar sobre isso?* [S.l.], 2016. Disponível em: ⟨https://www.inca.gov.br/sites/ufu.sti.inca.local/files//media/document//cartilha-cancer-de-mama-vamos-falar-sobre-isso2016.pdf, Visiton05-31-2020⟩.

[2] AICR. *American Institute for Cancer Research, Breast cancer statistics*. [S.l.], 2020. Disponível em: ⟨https://www.wcrf.org/dietandcancer/cancer-trends/breast-cancer-statistics,Visiton05-31-2020⟩.

[3] WHO. *World Health Organization, Breast cancer*. [S.l.], 2020. Disponível em: ⟨https://www.who.int/cancer/prevention/diagnosis-screening/breast-cancer/en/,Visiton05-31-2020⟩.

[4] BELSARE, A.; MUSHRIF, M. Histopathological image analysis using image processing techniques: An overview. *Signal Image Process Int J*, v. 3, 11 2011.

[5] Titoriya, A.; Sachdeva, S. Breast cancer histopathology image classification using AlexNet. In: *4th International Conference on Information Systems and Computer Networks (ISCON)*. [S.l.: s.n.], 2019. p. 708–712.

[6] WOLF, B.; SCHOLZE, C. "medicine 4.0". *Current Directions in Biomedical Engineering*, v. 3, n. 2, p. 183–186, 2017.

[7] Spanhol, F. A. et al. A dataset for breast cancer histopathological image classification. *IEEE Transactions on Biomedical Engineering*, v. 63, n. 7, p. 1455–1462, 2016.

[8] LECUN, Y. et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, v. 86, p. 2278 – 2324, 12 1998.

[9] Pereira Jr., P. C. et al. Comparison of classical computer vision vs.convolutional neural networks for weed mapping in aerial images. *Revista de Informática Teórica e Aplicada (RITA)*, v. 27, n. 4, p. 11–23, 12 2020.

[10] ZEMOURI, R.; ZERHOUNI, N.; RACOCEANU, D. Deep learning in the biomedical applications: Recent and future status. *Applied Sciences*, v. 9, n. 8, 2019.

[11] GULSHAN, V. et al. Development and Validation of a Deep Learning Algorithm for Detection of Diabetic Retinopathy in Retinal Fundus Photographs. *JAMA*, v. 316, n. 22, p. 2402–2410, 2016.

[12] GAYATHRI, S.; GOPI, V. P.; PALANISAMI, P. A lightweight cnn for diabetic retinopathy classification from fundus images. *Biomedical Signal Processing and Control*, v. 62, p. 102115, 2020.

[13] SU, Y.; LI, D.; CHEN, X. Lung nodule detection based on faster r-cnn framework. *Computer Methods and Programs in Biomedicine*, v. 200, p. 105866, 2021.

[14] Spanhol, F. A. et al. Breast cancer histopathological image classification using convolutional neural networks. In: *2016 International Joint Conference on Neural Networks (IJCNN)*. [S.l.: s.n.], 2016. p. 2560–2567.

[15] BEJNORDI, B. E. et al. Diagnostic Assessment of Deep Learning Algorithms for Detection of Lymph Node Metastases in Women With Breast Cancer. *JAMA*, v. 318, n. 22, p. 2199–2210, 2017.

[16] SILVA, D. C. S. e.; CORTES, O. A. C. On convolutiona neural networks and transfer learning for classifying breast cancer on histopahological images using gpu. In: *XXVII Brazilian COngress on Biomedical Engineering*. [S.l.: s.n.], 2020.

[17] Ismail, N. S.; Sovuthy, C. Breast cancer detection based on deep learning technique. In: *2019 International UNIMAS STEM 12th Engineering Conference (EnCon)*. [S.l.: s.n.], 2019. p. 89–92.

[18] Singh, R. et al. Imbalanced breast cancer classification using transfer learning. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, p. 83–93, 2020.

[19] SIMONYAN, K.; ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. In: *International Conference on Learning Representations*. [S.l.: s.n.], 2015.

[20] SHALLU; MEHRA, R. Breast cancer histology images classification: Training from scratch or transfer learning? *ICT Express*, v. 4, n. 4, p. 247–254, 2018.

[21] SAIKIA, A. R. et al. Comparative assessment of CNN architectures for classification of breast fnac images. *Tissue and Cell*, v. 57, p. 8–14, 2019. EM in cell and tissues.

[22] SITAULA, C.; HOSSAIN, M. Attention-based VGG-16 model for covid-19 chest x-ray image classification. *Applied Intelligence*, Springer, v. 51, p. 2850–2863, 2020.

[23] KHAMPARIA, A. et al. Diagnosis of breast cancer based on modern mammography using hybrid transfer learning. *Multidimensional Systems and Signal Processing*, v. 32, p. 747–765, 2021.

[24] II, T. B. *Introduction to Deep Learning Using R: a step-by-step guide to learning and implementing Deep Learning Models Using R*. [S.l.]: Apress, 2017.

[25] TORREY, L.; SHAVLIK, J. Transfer learning. In: OLIVAS, E. S. et al. (Ed.). *Handbook Of Research On Machine Learning Applications and Trends: Algorithms, Methods and Techniques*. , Hershey, New York: Information Science Reference, 2009. cap. 11, p. 242–264.

[26] RANZATO, M. et al. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In: *2007 IEEE Conference on Computer Vision and Pattern Recognition*. [S.l.: s.n.], 2007. p. 1–8.

[27] LIN, M.; CHEN, Q.; YAN, S. *Network In Network*. 2014.

[28] ABADI, M. et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. 2015. Software available from tensorflow.org. Disponível em: ⟨https://www.tensorflow.org/⟩.

[29] PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011.

[30] Google. *Kaggle platform*. [S.l.], 2021. Disponível em: ⟨https://www.kaggle.com/,Visitedon27-09-2021⟩.

[31] KINGMA, D.; BA, J. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[32] GLOROT, X.; BENGIO, Y. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research - Proceedings Track*, v. 9, p. 249–256, 01 2010.

[33] BEHN, R. Why measure performance? different purposes require different measures. *Public Administration Review*, v. 63, p. 586 – 606, 09 2003.

[34] SELVARAJU, R. R. et al. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *CoRR*, abs/1610.02391, 2016. Disponível em: ⟨http://arxiv.org/abs/1610.02391⟩.

[35] WOO, S. et al. CBAM: convolutional block attention module. *CoRR*, abs/1807.06521, 2018. Disponível em: ⟨http://arxiv.org/abs/1807.06521⟩.

[36] NAIR, V.; HINTON, G. Rectified linear units improve restricted boltzmann machines vinod nair. In: . [S.l.: s.n.], 2010. v. 27, p. 807–814.