Non-invasive, Early Detection of Invasive Ductal Carcinoma (IDC) via Deep Convolutional Neural Networks Using Breast Cancer Histology Images

Aniruddha Murali

Abstract— Breast cancer is responsible for causing the greatest number of cancer-related deaths among women, impacting 2.1 million women every year (WHO). Breast cancer is the most common form of cancer in women, with invasive ductal carcinoma (IDC) representing 80 percent of all breast cancer diagnoses. One way to reduce the number of deaths caused by breast cancer is to perform early diagnosis to detect the presence of a malignant tumor before the tumor gets too harmful. While there are several methods of diagnosing and testing a tumor, they all have their own sets of problems: they are time-consuming, expensive, and limited in their ability to diagnose a variety of tumors. Accurately identifying and categorizing breast cancer subtypes is an important clinical task, and automated methods can be used to save time and reduce error, potentially saving many lives. This study presents a deep learning approach for automatic detection and visual analysis of IDC tissue regions in whole slide images of breast cancer. The approach is similar to how the human brain uses different interpretation levels or layers of most representative and useful features, resulting in a hierarchical learned representation. These methods have been shown to outpace traditional approaches of most challenging problems in several areas such as speech recognition and object detection. The deep learning framework used were convolutional neural networks (CNNs). The model utilizes the Keras library to create the convolutional neural network and a dataset from Case Western Reserve University consisting of over 270,000 images of patches of breast cancer specimens. The model achieved 88-90% accuracy.

Index Terms— breast cancer, breast cancer diagnoses, invasive ductal carcinoma, idc, deep learning, machine learning, artificial intelligence, prediction, malignancy, convolutional neural networks.

IJSER

1 Introduction

Breast cancer has been a prominent issue among women; it has been very difficult to determine whether a breast tumor is malignant or not without being invasive and to address the cancer before it becomes too problematic. Cancers can result from alterations in genes encoding cellular signaling molecules, especially protein kinases (My Cancer Genome 2016). Types of gene alterations that can result in cancers include: single nucleotide variants (point mutations), small duplications of consecutive nucleotides, insertions or deletions involving one or a few nucleotides, changes in exon or gene copy numbers, and structural variants in genetic material including translocations and inversions (My Cancer Genome 2016).

Invasive ductal carcinoma is the most common type of breast cancer. About 80% of all breast cancers are invasive ductal carcinomas. According to the American Cancer Society, more than 180,000 women in the United States find out they have invasive breast cancer each year. Most of them are diagnosed with invasive ductal carcinoma.

Current methods of tumor testing are capable of identifying mutations in tumor DNA. However, these methods generally come with a set of drawbacks. Almost all current methods of tumor testing can only detect a specific mutation; other mutations that may be present in tumor DNA cannot be detected (My Cancer Genome 2016). Such methods are limited in the types or number of mutations that can be detected in tumor DNA. They can also be labor intensive and/or expensive, often involving the use of highly sophisticated technology. Completion of diagnosis can take anywhere from 2-3 days to several weeks, and such tests also have a possibility of false negatives and false positives (My Cancer Genome 2016).

Early diagnosis of breast tumors can help doctors to provide a mostly accurate assessment of a breast tumor to their patients. The most common method of diagnosing breast tumors is mammography, which is an x-ray imaging method used to examine a breast for early detection of breast cancer (NIBIB 2015). A radiologist examines a mammogram to identify any potential abnormalities in the breast. Mammography has been shown to reduce breast cancer mortality by about 20% in high-resource settings (WHO 2018). However, a mammogram is examined only by a radiologist. If it is not clear that the tumor is malignant or benign, there is a chance that the radiologist could give an inaccurate result. In one study, 100 mammograms were submitted to nine radiologists. The diagnosis or suspicion of cancer varied from 10-55% (Devitt 2016). The denser the breast, the more difficult it is to produce an image, so it will be harder to diagnose (NIBIB 2015).

Machine learning algorithms such as decision trees, logistic regression, and support vector machines have been common choices for creating models that can be used to classify the malignancy of breast tumors. Previous studies have shown that they can demonstrate over 90% accuracy in cancer diagnosis, much more than today's standard of 80% accuracy. However, the data that these algorithms use are collected by humans. Humans choose, for example, the boundaries of a feature in a tumor; they could be missing many pixels or be including extra pixels for each feature, meaning that the data itself may not be accurate. While these algorithms do generally yield high accuracy, they do not take into account the bias from human error. With this important source of bias, the validity of such models are questionable.

The recent increase in available computing power and dataset size, along with the ability to examine images directly, has made convolutional neural networks (CNNs) popular for image classification problems. Contrary to the traditional approach of hand-crafted feature extraction methods, CNNs learn useful features directly from the training image patches by the optimization of the classification loss function. These deep learning models have achieved excellent performance in image classification challenges in different fields, including medical image analysis. CNNs reduce the field-knowledge needed to design a classification system. Because of this, the performance of the model is less biased by the dataset used and similar network architectures can achieve good results on different problems.

2 Methods

2.1 Dataset

The original dataset from Case Western University consists of 162 whole mount slide images of Breast Cancer (BCa) specimens scanned at 40x. From that, 277,524 patches of size 50 x 50 were extracted (198,738 IDC negative and 78,786 IDC positive). Each patch's file name is of the format: u_xX_yY_classC.png, where u is the patient ID, X is the x-coordinate of where this patch was cropped from, Y is the y-coordinate of where this patch was cropped from, and C indicates the class, where 0 is non-IDC and 1 is IDC.

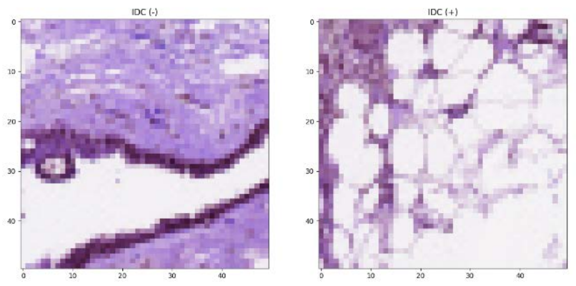


Figure 1: One sample each of a non-IDC tumor and an IDC tumor from the dataset.

• *Aniruddha Murali is actively working on scientific research in the field of artificial intelligence as a senior at Staples High School, Westport, CT, USA, PH-12034518797. E-mail: aniruddha.murali@gmail.com*

2.2 Data Preparation

The dataset of images was split into training and testing sets. The training set consisted of 90% of the dataset while the testing set comprised 10% of the dataset. Classes were retrieved from the file name. In order to eliminate the bias of one class over the other, the training set was normalized so that half of the images in the training set were IDC+ and the other half were IDC-.

2.3 CNN Architecture

CNNs are comprised of three types of layers: convolutional layers, pooling layers and fully-connected layers. When these layers are stacked, a CNN architecture has been formed. The input layer of the CNN will hold the pixel values of the inputted image. The convolutional layer will determine the output of neurons of which are connected to local regions of the input through the calculation of the scalar product between their weights and the region connected to the input volume. The rectified linear unit (ReLu) applies an activation function such as the sigmoid function to the output of the activation produced by the previous layer; this keeps the value of the neurons between 0 and 1. The pooling layer will then simply perform downsampling along the spatial dimensionality of the given input, further reducing the number of parameters within that activation. The fully-connected layers will then perform the same duties found in standard neural networks and attempt to produce class scores from the activations to be used for classification of the input. ReLu may be used between these layers to improve performance.

2.4 Convolutional Layers

Convolutional layers focus on the use of filters (also referred to as kernels). Filters are convolved with the image. The region that the filter is applied over is called the receptive field. The filters are arrays of numbers called weights. When convolving the filters with the image, the pixel values of the image are multiplied by the weights of the filter and then added together to produce a scalar number called the weighted sum, which is stored in an activation map. The scalar number produced is in the position of the activation map that corresponds with receptive field.

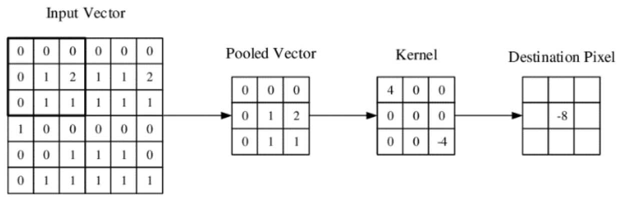


Figure 2: A visual representation of a convolutional layer. The center element of the kernel is placed over the input vector, of which is then calculated and replaced with a weighted sum of itself and any nearby pixels.

Convolutional layers are also able to significantly reduce the complexity of the model through the optimisation of its

IJSER

output. These are optimised through three hyperparameters: the depth, the stride, and zero-padding. The depth of the output volume produced by the convolutional layers can be manually set through the number of neurons within the layer to a the same region of the input. Reducing this hyperparameter can significantly minimise the total number of neurons of the network, but it can also significantly reduce the pattern recognition capabilities of the model. The stride can be defined in which the depth is set around the spatial dimensionality of the input in order to place the receptive field. For example, if we were to set a stride as 1, then we would have a heavily overlapped receptive field producing extremely large activations. Alternatively, setting the stride to a greater number will reduce the amount of overlapping and produce an output of lower spatial dimensions. Zero-padding is the simple process of padding the border of the input and is an effective method to give further control as to the dimensionality of the output volumes. It is important to understand that through using these techniques, we will alter the spatial dimensionality of the convolutional layers output. To calculate this, you can make use of the following formula:

$$\frac{(V - R) + 2Z}{S + 1}$$

V represents the input volume size (height×width×depth), R represents the receptive field size, Z is the amount of zero padding set, and S referring to the stride. If the calculated result from this equation is not equal to a whole integer then the stride has been incorrectly set, as the neurons will be unable to fit neatly across the given input.

IJSER

2.5 Pooling Layers

Pooling layers aim to gradually reduce the dimensionality of the representation, and thus further reduce the number of parameters and the computational complexity of the model. Pooling layer operates on each activation map independently. The most common approach used in pooling is max pooling.

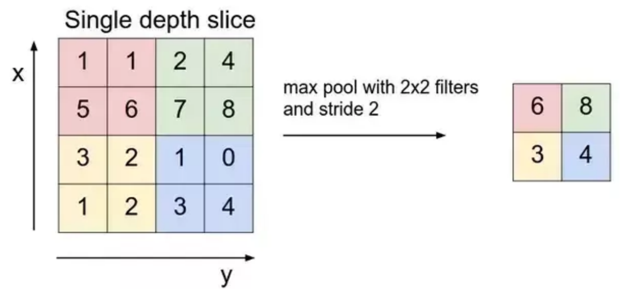


Figure 3: Example of max pooling with 2x2 filters and a stride of 2

2.6 Fully-connected Layers

Fully-connected layers take an input volume (whatever the output is of the convolutional layer or ReLu or pooling layer preceding it) and outputs an N dimensional vector where N is the number of classes that the program has to choose from. In the case of classifying IDC, N = 2 since either a tumor shows positive or negative for IDC. Each number in this N dimensional vector represents the probability of a certain class. The fully-connected layer looks at the output of the previous layer (which should represent the activation maps of high-level features) and determines which features correlate the most to a particular class. This correlation is determined using particular weights so that when computing the product of the weights and the values in the neurons of the previous layer, the correct probabilities for the different classes are calculated. These probabilities can be used to classify the image: whichever class has the highest probability would be the returned classification of the CNN.

2.7 Backpropagation

The computer is able to adjust its filter values (or weights) is through a training process called backpropagation. Before the CNN starts, the weights or filter values are randomized. The filters don't know to look for edges and curves. Backpropagation can be separated into 4 distinct sections: the forward pass, the loss function, the backward pass, and the weight update. During the forward pass, you take a training image and pass it through the whole network. At first, the neural network, with its current weights, isn't able to look for low level features and therefore cannot make any reasonable conclusions about what the classification is. This goes to the loss function part of backpropagation. While there are many different functions used to calculate loss, all of them convey how often the neural network is wrong in its classification. The formula used for the loss function will be explained later on. It is expected that the loss will be extremely high for the first couple of training images because of the lack of inputted observations. The goal is to get to a point where the predicted label from the CNN is the same as the training label. To accomplish this, loss has to be minimized. Visualizing this as just an optimization problem in calculus, we want to find out which inputs (weights in our case) most directly contributed to the loss (or error) of the network.

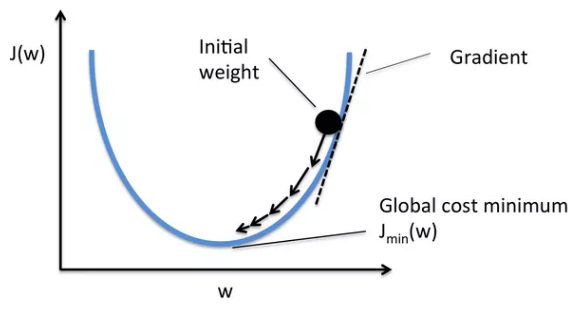


Figure 4: Illustration of minimizing the loss function. Gradient function is used to eventually reach the global cost minimum; this is the point at which loss is minimized.

This is the mathematical equivalent of a dL/dW (derivative of L(W)) where W are the weights at a particular layer and L is the loss at each of those weights. Once loss is calculated, the neural network will perform a backward pass through the network in which it determines which weights contributed the most to the loss and find ways to adjust them so that the loss decreases. Once this derivative is computed, the neural network will update the weights. This is where the weights of the filters are updated so that they change in the opposite direction of the gradient. The process of forward pass, loss function, backward pass, and parameter update is one training iteration called an epoch. The program will repeat this process for a fixed number of iterations for a training set of images. By inputting many training images and repeating the process of backpropagation, the CNN should be trained well enough so that the weights of the layers are tuned correctly.

2.8 Loss Function

Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label. A perfect model would have a log loss of 0. As the predicted probability approaches 1, log loss slowly decreases. As the predicted probability decreases, however, the log loss increases rapidly.

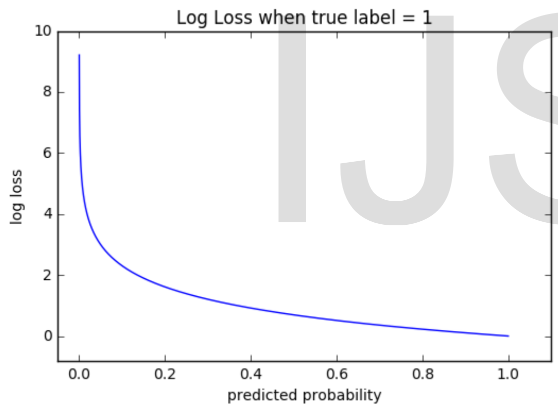


Figure 5: Example of a cross entropy (log loss) graph.

Log loss penalizes both types of errors, but especially the predictions that are confident and wrong. In binary classification, cross-entropy can be calculated using the formula

$$(y\log(p) + (1-y)\log(1-p)),$$

where y is the actual class label of the observation and p is the predicted class label of the same observation.

2.9 Dropout

Dropout is a neural network regularization technique in which randomly selected neurons are ignored during training and are not used when evaluating the model. When the neurons are randomly dropped out of the network during the training, the other neurons will have to step in and handle the representation required to make a prediction for the missing neurons. The effect is that the network becomes less sensitive to the specific weights of neurons, resulting in a network that is capable of better generalization and is less likely to overfit the training data. Dropout is implemented by randomly selecting neurons to be dropped-out with a given probability (e.g. 20%) each weight update cycle. The dropout technique can be implemented on a visible input layer or hidden layer.

2.10 Activation Functions

2.10.1 Signmoid

The sigmoid function takes any real number and returns a standardized value that falls between 0 and 1. It's often used in logistic regression for a binary classification and also as activation function in neural networks. The output of the function returns the probability of an event occurring based on the sigmoid distribution. The sigmoid function is the following:

$$P(x) = \frac{1}{1+e^{-x}}$$

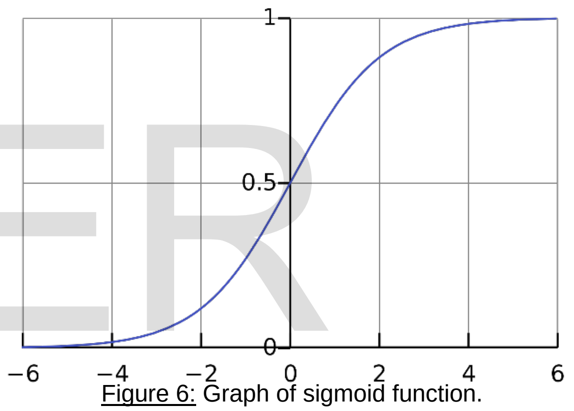


Figure 6: Graph of sigmoid function.

2.10.2 Softmax

The softmax function is a generalization of the sigmoid function used to handle multi-class classification. Softmax provides the probability of each class occurring. The output with highest probability is the predicted classification value. The sum of returned values of the softmax function is always equal to 1. It is proven that sigmoid is a particular case of softmax with i=2. The softmax function is often used in deep learning when working with neural networks and can be used to classify images. The equation use in softmax is:

$$S(y_i) = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}}$$

2.10.3 ReLu (Rectified Linear Unit)

ReLu is the most commonly used activation function in deep learning algorithms. ReLu is half rectified from the bottom. f(x) is zero when x is less than zero and f(x) is equal to x when x is greater than or equal to zero.

$$f(x) = \begin{cases} 0 & \text{for} \quad x < 0 \\ x & \text{for} \quad x \geq 0 \end{cases}$$

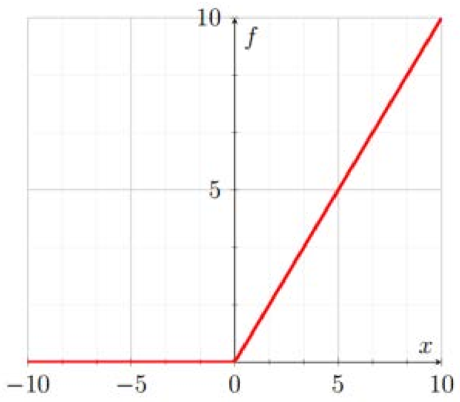


Figure 7: Graph and function of ReLu activation function

2.11 Confusion Matrix

A confusion matrix is a technique for summarizing the performance of a classification algorithm. Classification accuracy alone can be misleading if there is an unequal number of observations in each class or if there are more than two classes in the dataset. Calculating a confusion matrix can provide a better idea of what the classification model is getting right and what types of errors it is making. The confusion matrix shows the ways in which the classification model is confused when it makes predictions. The number of correct and incorrect predictions are summarized with count values and broken down by each class.

CONFUSION MATRIX	Predicted: Negative	Predicted: Positive
Actual: Negative	True Negative(TN)	False Positive (FP)
Actual: Positive	False Negative (FN)	True Positive (TP)

Figure 8: Confusion matrix table illustrating different types of values.

The confusion matrix provides the number of data points that fit each of the following four categories:

- "True positive" for correctly predicted event values.
- "False positive" for incorrectly predicted event values.
- "True negative" for correctly predicted no-event values.
- "False negative" for incorrectly predicted no-event values.

False positive values illustrate a Type I error while false negative values illustrate a Type II error. These four values can help in calculating more advanced classification metrics:

Accuracy: Overall, how often is the model correct?
(TP+TN)/Total

Misclassification/Error Rate: Overall, how often is the model wrong?
(FP+FN)/Total

True Positive Rate (Sensitivity/Recall): When the result is actually positive, how often does the model correctly predict positive?
TP/(TP+FN)

False Positive Rate: When the result is actually negative, how often does the model incorrectly predict positive?
FP/(TN+FP)

True Negative Rate (Specificity): When the result is actually negative, how often does the model correctly predict negative?
TN/(TN+FP)

False Negative Rate: When the result is actually positive, how often does the model incorrectly predict negative?
FN/(TP+FN)

Precision: When the model predicts positive, how often is it correct?
TP/(TP+FP)

Prevalence: How often does the positive condition actually occur in our sample?
(FN+TP)/Total

IJSER

3 Results

3.1 Implementation of CNN Architecture

The Sequential CNN architecture used is shown through the following code and steps:

```
model.add(Conv2D(32, kernel_size=(3, 3), activation='relu',
input_shape=input_shape, strides=e))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(num_classes, activation='softmax'))
```

1. The first hidden layer is a convolutional layer called a Conv2D. This layer has 32 features maps with 3x3 kernels and a ReLu activation function. Input image is 50-pixel x 50-pixel x 3 in shape. Strides window is 2.

2. The 2nd hidden layer is convolution layer called a Conv2D. This layer has 64 feature maps with size of 3 x 3 and ReLu activation function.

3. The 3rd hidden layer is a pooling layer that takes the maximum value called MaxPooling2D. It is configured with a pool size of 2 x2.

4. The 4th hidden layer is a regularization layer using dropout called Dropout. It is configured to randomly drop 25% of neurons in the layer in order to reduce overfitting.

5. The 5th hidden layer is Flatten. This layer converts the 2D matrix data to a vector and allows the output to be processed by standard fully connected layers.

6. The 6th hidden layer is a fully-connected layer with 128 neurons and the ReLu activation function is used.

7. The 7th hidden layer is a Dropout regularization layer. It is configured to randomly drop 50% of neurons in the layer in order to reduce overfitting.

8. Finally, the output layer has two neurons for the 2 classes and the softmax activation function to output the prediction with probability for each class.

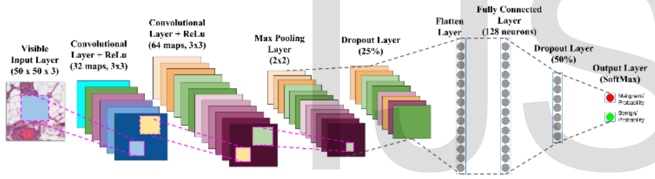

Visible Input Layer (50 x 50 x 3)
Convolutional Layer + ReLu (32 maps, 3x3)
Convolutional Layer + ReLu (64 maps, 3x3)
Max Pooling Layer (2x2)
Dropout Layer (25%)
Flatten Layer
Fully Connected Layer (128 neurons)
Dropout Layer (50%)
Output Layer (SoftMax)
Malignant/ Probability
Benign/ Probability

Figure 9: Sequential CNN architecture implemented in deep learning program.

3.2 Training vs. Validation

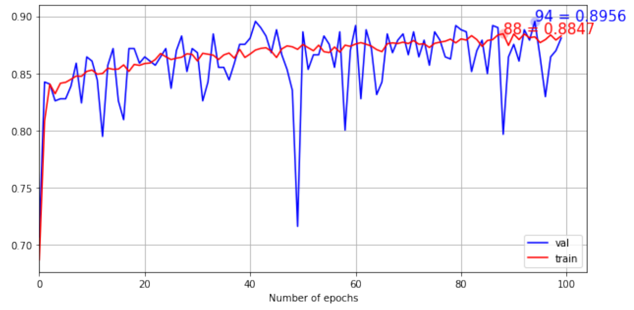


Figure 10: Training and validation curves representing accuracy over 100 epochs

Training data comprised 90% of the entire dataset while testing data comprised the other 10%. When training the CNN up to 100 epochs, the model achieved its maximum accuracy after 88 epochs with 88.47% accuracy. When validating the CNN using the testing set up to 100 epochs, the model achieved maximum accuracy after 94 epochs with 89.56% accuracy. The training and validation curves grew with each other for the majority of the epochs, although there were large discrepancies after about 48 epochs. The model seems to be accurate and reliable.

3.2 Accuracy and Loss

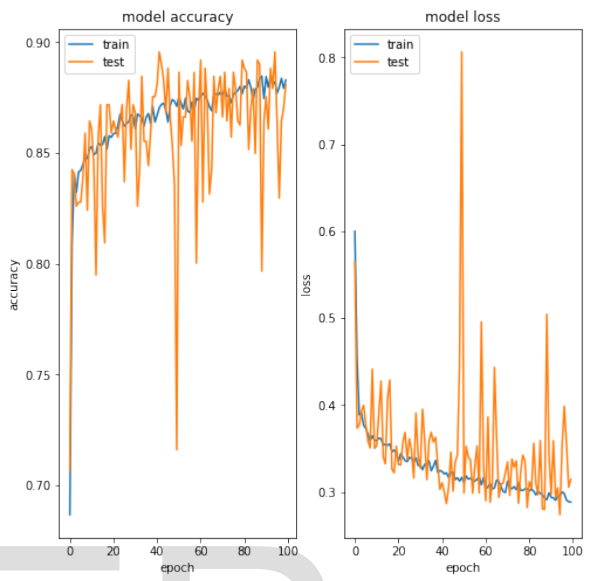
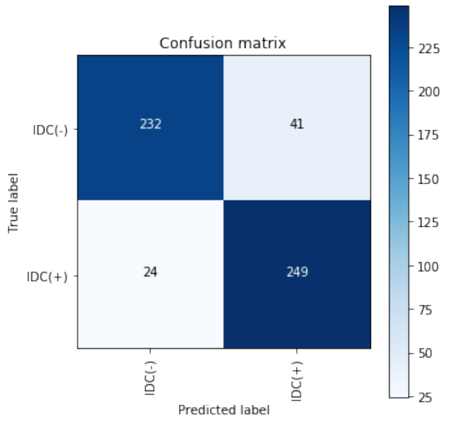


Figure 11: Model accuracy and model loss up to 100 epochs

Model accuracy was calculated by determining the number of correct predictions divided by the total number of predictions made. The explanation on model accuracy was provided in Section 3.2. Cross-entropy loss was used for the loss function. In this loss function, errors are penalized severely, as shown after about 48 epochs. However, the general trend showed model loss to decrease as the number of epochs increased. Testing loss dropped along with training loss, again indicating that the model seems to be accurate and reliable.

3.2 Confusion Matrix



	IDC(-)	IDC(+)
IDC(-)	232	41
IDC(+)	24	249

Figure 12: Resulting confusion matrix for IDC classification. Total = 546, TP = 232, TN = 249, FP = 24, FN = 41

This is the list of rates that are often computed from a confusion matrix for a binary classifier:

Accuracy:
(TP+TN)/Total = (232+249)/546 = 0.88

Misclassification/Error Rate:
(FP+FN)/Total = (24+41)/546 = 0.12

True Positive Rate (Sensitivity/Recall):
TP/(TP+FN) = 232/273 = 0.85

False Positive Rate:
FP/(TN+FP) = 10/60 = 0.17

True Negative Rate (Specificity):
TN/(TN+FP) = 50/60 = 0.83

False Negative Rate:
FN/(TP+FN) = 41/273 = 0.15

Precision:
TP/(TP+FP)= 232/256 = 0.91

Prevalence:
(FN+TP)/total = 273/546 = 0.50

The confusion matrix shows that the classifying model achieved high accuracy (88%) and precision (91%) and that it achieved relatively high sensitivity (85%) and specificity (83%). The error rate (12%) was low, and the false positive (17%) and false negative (15%) rates were relatively low.

The confusion matrix shows that it predicted positive 232 + 24 = 256 times and predicted negative 249 + 41 = 290 times. Prevalence was 50% and the number of times the model predicted positive was close to the number of times it predicted negative, so it is determined that the model did not show bias towards a specific class.

4 Implications

This Sequential CNN model can be used to detect invasive ductal carcinoma in patients. Since this is only a prediction, this should not be the only means of determining if a breast tumor is malignant or benign; however, it can give doctors a good sense of the severity of the tumor is harmful or not. If used at an early stage, the diagnosis from this model can help doctors take immediate action to address any possible malignant tumors. As there are 2.1 million women affected by breast cancer every year, this technology can help thousands of women get early treatment. This model achieves between 86-90% accuracy, which is better than today's standard of 80% accuracy. Moreover, the breast cancer histology image is directly inputted into the model without human interference, so human error in the diagnosis is minimized.

One possible next step is to introduce a learning rate variable. Learning rate is a parameter that is chosen by the programmer. A high learning rate means that bigger steps are taken in the weight updates and thus, it may take less time for the model to converge on an optimal set of weights. However, a learning rate that is too high could result in jumps that are too large and not precise enough to reach the optimal point. If the right learning rate is chosen, the model will be able to efficiently learn from a dataset while achieving high accuracy.

It is important to note that due to the lack of computer processing power available, more complex CNN architectures could not be implemented. If there was access to powerful graphics processing units (GPUs), more sophisticated architectures that utilized more convolutional, pooling, and fully-connected layers could be tested, potentially achieving significantly higher accuracy compared to the 88-90% accuracy achieved in this study. However, despite the lack of a powerful GPU, this study managed to achieve a higher accuracy than current methods of breast cancer diagnosis, demonstrating the potential of applications of artificial intelligence in healthcare for diagnosing diseases with higher accuracy at earlier stages.

IJSER

Acknowledgment

The author would like to thank Mrs. Karen Thompson for mentoring him on the overall research process.

References

[1] Janowczyk, Andrew, and Anant Madabhushi. "Deep Learning for Digital Pathology Image Analysis: A Comprehensive Tutorial with Selected Use Cases." Journal of Pathology Informatics, vol. 7, no. 1, 2016, p. 29., doi:10.4103/2153-3539.186902.

[2] Liu, Yun, et al. "Artificial Intelligence–Based Breast Cancer Nodal Metastasis Detection." Archives of Pathology & Laboratory Medicine, 2018, doi:10.5858/arpa.2018-0147-oa.

[3] Bardou, Dalal, et al. "Classification of Breast Cancer Based on Histology Images Using Convolutional Neural Networks." IEEE Access, vol. 6, 1 June 2018, pp. 24680–24693., doi:10.1109/access.2018.2831280.

[4] Krizhevsky, Alex, et al. "ImageNet Classification with Deep Convolutional Neural Networks." Communications of the ACM, vol. 60, no. 6, 2017, pp. 84–90., doi:10.1145/3065386.

[5] Charan, Saira, et al. "Breast Cancer Detection in Mammograms Using Convolutional Neural Network." 2018 International Conference on Computing, Mathematics and Engineering Technologies (ICoMET), Mar. 2018, doi:10.1109/icomet.2018.8346384.

[6] Chougrad, Hiba, et al. "Deep Convolutional Neural Networks for Breast Cancer Screening." Computer Methods and Programs in Biomedicine, vol. 157, 29 Nov. 2017, pp. 19–30., doi:10.1016/j.cmpb.2018.01.011.

[7] Rakhlin, Alexander, et al. "Deep Convolutional Neural Networks for Breast Cancer Histology Image Analysis." Lecture Notes in Comput-

er Science Image Analysis and Recognition, 3 Apr. 2018, pp. 737–744., doi:10.1007/978-3-319-93000-8_83.
[8] O'Shea, Keiron, and Ryan Nash. "An Introduction to Convolutional Neural Networks." Nov. 2015, pp. 1–11.
[9] Early Diagnosis and Screening - Breast Cancer. (n.d.). Retrieved from World Health Organization website: http://www.who.int/cancer/prevention/diagnosis-screening/breast-cancer/en/
[10] Vnencak-Jones, C., M. Berger, W. Pao. 2016. Types of Molecular Tumor Testing. My Cancer Genome: https://www.mycancergenome.org/content/molecular-medicine/types-of-molecular-tumor-testing/
[11] Mammography. (n.d.). Retrieved from National Institute of Biomedical Imaging and Bioengineering website: https://www.nibib.nih.gov/science-education/science-topics/mammography
[12] Deshpande, Adit. "A Beginner's Guide To Understanding Convolutional Neural Networks." A Beginner's Guide To Understanding Convolutional Neural Networks – Adit Deshpande – CS Undergrad at UCLA ('19), 20 July 2016, adeshpande3.github.io/adeshpande3.github.io/A-Beginner's-Guide-To-Understanding-Convolutional-Neural-Networks/.
[13] "Limitations of Mammograms | How Often Are Mammograms Wrong?" American Cancer Society, American Cancer Society, 9 Oct. 2017, www.cancer.org/cancer/breast-cancer/screening-tests-and-early-detection/mammograms/lImitations-of-mammograms.html.
[14] "Convolutional Neural Networks (CNNs / ConvNets)." CS231n Convolutional Neural Networks for Visual Recognition, Stanford University, https://cs231n.github.io/convolutional-networks/.
[15] "Invasive Ductal Carcinoma: Diagnosis, Treatment, and More." Breastcancer.org, Breastcancer.org, 16 Oct. 2018, www.breastcancer.org/symptoms/types/idc.
[16] Srivastava, Nitish, et al. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting." Journal of Machine Learning Research, Edited by Yoshua Bengio, vol. 15, 2014, pp. 1929–1958.

IJSER