Journal of Applied Statistics & Machine Learning
Volume 2; Number 1; 2023 : pp. 1-37

BREAST CANCER IN HISTOPATHOLOGICAL DATA THROUGH IMAGE CLASSIFICATION USING DEEP LEARNING METHODS

Vikash Sharma[1], Siddhartha Roy[2] and Girdhar G. Agarwal[3]

[1]Independent Consultant in Analytics, Data Science, Machine Learning and AI, Bengaluru, India,
E-mail: vikash.sharma.inbox@gmail.com (corresponding author)
[2]Associate Director (Advanced Analytics) Gainwell Technologies, Bengaluru, India
E-mail: arpsid@gmail.com
[3] Ex-Professor, Department of Statistics, Lucknow University, Lucknow, India
E-mail: girdhar1751@gmail.com

ARTICLEINFO

Received: 26 December 2022
Revised: 05 January 2023
Accepted: 11 February 2023
Online: 10 March 2023

To cite this paper:
Vikash Sharma, Siddhartha Roy & Girdhar G. Agarwal (2023). Breast Cancer in Histopathological Data through Image Classification using Deep Learning Methods. *Journal of Applied Statistics & Machine Learning*. 2(1): pp. 1-37.

ABSTRACT

Breast cancer is one of the most common cancers afflicting women. Early detection and effective treatment are critical to improving the chances of survival. Since invasive ductal carcinoma (IDC) accounts for 80% of all breast cancers, early detection of IDC cells plays an instrumental role in controlling cancer outcomes. While histopathological image analysis is the gold standard for detecting cancer, it is very challenging for pathologists to examine large patches of benign regions for identifying malignant cells. This process is not only prone to pathologists' subjectivity but also quite time-consuming, laborious and expensive. Deep learning techniques, particularly convolutional neural networks (CNNs), can mechanize the detection process to make it more objective, precise, and faster since they are good at learning predominant features automatically. However, lack of enough labelled and class balanced data samples are some of the practical challenges in adoption of deep learning methods for such problems. In this paper, we propose an image classification model using CNNs for IDC cell detection in histopathology slides. Further, we have performed a comparative analysis of some of the state-of-the-art CNN architectures and applied transfer learning techniques. By trying out experiments on such kinds of models through transfer learning and optimization techniques, we have identified the most suitable transfer learning approach based on the EfficientNet-B7 network that has achieved accuracy of 90%, sensitivity of 91%, specificity of 90%, F1-score of 84% and balanced accuracy of 91%. This is an improvement on some of the previous research literature on this dataset. Through our approach, this research topic has focused on the benefits of using

CNNs and transfer learning techniques to solve the IDC cell image classification problem with better accuracy and efficiency. This helps us in laying down a state-of-the-art approach for IDC detection through breast cancer histopathology image classification.

Keywords: deep learning, CNN architecture, accuracy, precision, transfer learning, hyperparameter tuning, learning rate

INTRODUCTION

1.1 Background of the Study

As per The World Health Organization (WHO, 2020), breast cancer is the most common cancer afflicting women. It is estimated that in 2018 alone, breast cancer has contributed to 2.1 million cases and 627,000 deaths. Further, breast cancer represents 24% of all cancers diagnosed in women and accounts for the largest proportion of cancer-related deaths (approximately 15%) in women.

Early diagnosis and effective treatment are instrumental in controlling cancer mortality. Due to advancements in detection techniques and earlier diagnosis, there has been a decrease in premature mortality rates (WHO, 2020). This drives the need to improve detection techniques at an early stage. Despite advances in medical automation, histopathological slide image diagnosis is still considered the gold standard for detecting cancer. However, this task is complex, time-consuming, laborious, expensive, and dependent on manual qualitative analysis of the pathologist. In less developed areas, there is a shortage of competent pathologists and the highly elaborate process adds to pathologist fatigue (Yan *et al.*, 2020). This is the key motivation to develop automatic methods that can not only enable the pathologist to enhance the effectiveness of the diagnostic process but also make the early detection faster and precise.

The most common types of breast cancer include invasive ductal carcinoma (IDC) and invasive lobular carcinoma. IDC accounts for approximately 80% of all breast cancers (BREASTCANCER.ORG, 2020). Detection of IDC cells through histopathology images is a laborious and challenging work since the pathologist needs to examine large tissue patches of benign regions to eventually identify the malignant areas (Cruz-Roa *et al.*, 2014). This is where deep learning methods, particularly convolutional neural networks (CNNs), are very good at predicting the classification of IDC cell images by focusing on predominant features in an automated way.

In related research on artificial intelligence applications in breast cancer histopathological detection since 2012 (Zhou *et al.*, 2020), CNNs have been the most common method in image classification due to their ability to automatically learn features, the availability of publicly labelled datasets in recent years, the emergence of high-performance GPU computing and learnings from applications in natural language processing, and image recognition.

1.2 Literature Review

In the last four years, one of the most popular strategies to train CNNs has been "transfer learning" by either fine-tuning the parameters of a pre-trained state-of-the-art network as per the target task or training a new classifier on features extracted using a pre-trained network. Even though there are not many readily available public datasets on breast cancer histology, four datasets have been covered a lot in the research papers over the last few years. There is a strong adoption of transfer learning methods in a lot of papers for these datasets.

For the **BreakHis** dataset released by Spanhol *et. al.* in (Spanhol *et al.*, 2016) which has 7,909 histopathological images distributed across 40x, 100x, 200x, and 400x magnifications, Saxena *et. al.* proposed pre-trained ResNet50 and the kernelized weighted extreme learning machine to address the class imbalance problem in (Saxena *et al.*, 2020). They outperformed state-of-the-art models on 100x, 200x, and 400x magnifications using identical training-testing folds to deliver accuracy of 87.14%, 90.02%, and 84.16% respectively. In (Xu *et al.*, 2019), an innovative three-step attention based approach involving Partially Observed Markov Decision Process (POMDP) along with a combination of deep learning techniques of recurrent neural networks (RNNs), long short-term memory networks (LSTMs), and reinforcement learning was used. In (Jiang *et al.*, 2019), a smaller Squeeze-and-Excitation-Resnet module which provides better performance with fewer parameters than that used by a network of residual module and Squeeze-and-Excitation block (Hu *et al.*, 2020) is proposed. This model provides accuracy between 98.9% and 99.3% for binary classification, and between 90.7% and 93.8% for multi-class classification.

In (Aresta *et al.*, 2019), a dataset of 400 images (of dimension 2048 × 1536 pixels) stained with Hematoxylin and Eosin (H&E) was released under the Grand Challenge on Breast Cancer Histology (**BACH**) organized along with the 15th International Conference on Image Analysis and Recognition (ICIAR 2018). The goal was to classify them in four classes including normal,

benign, in-situ carcinoma and invasive carcinoma. In (Golatkar *et al.*, 2018), a patch selection method using nuclear density followed by a transfer learning based algorithm based on fine-tuning Inception-V3 network was proposed. This method achieved 85% accuracy over the four categories and 93% accuracy for non-cancer (normal/benign) over malignant (in-situ/ invasive carcinoma), which outperformed older benchmark. In (Ferreira *et al.*, 2018), they adopted a transfer learning approach based on fine-tuning and adding top layers to the Inception ResNet-V2 (Szegedy *et al.*, 2016) architecture, finally achieving 93% validation accuracy and 76% test accuracy. In (Kohl *et al.*, 2018), three transfer learning models based on VGG-19 (Zisserman *et al.*, 2015), Inception-v3 (Szegedy *et al.*, 2015) and DenseNet-161(Huang *et al.*, 2018) were used for the classification task. While these networks were pre-trained on ImageNet, DenseNet-161 outperformed other methods by achieving 94% accuracy. In (Wang *et al.*, 2018), one model using only transfer learning with VGG-16 (Zisserman *et al.*, 2015) architecture was compared with another model using transfer learning followed by a support vector machine (SVM) classifier, achieving 91.7% accuracy on the test set.

The **Camelyon16** dataset has 400 whole-slide images (split into 270 for training and 130 for testing) based on samples from Radbound UMC and UMC Utrecht, released as a part of "Camelyon Grand Challenge" for detection of metastatic breast cancer in WSIs of sentinel lymph node biopsies. In (Wang *et al.*, 2016), an approach using GoogLeNet architecture was used to secure AUC of 92.5%. A human pathologist had independently reviewed this as well to obtain AUC of 96.6%. Combining both these approaches resulted in AUC of 99.5% which was an 85% reduction in human misclassification rate. In (BenTaieb *et al.*, 2017), a recurrent visual attention-based architecture was used on a similar setup to achieve AUC of 96%. In (Lin *et al.*, 2018), a fast and dense screening approach called ScanNet was introduced, achieving AUC of 98.75%. This was built on the VGG-16 network architecture by replacing the last 3 fully connected layers with fully convolutional layers.

In (Cruz-Roa *et al.*, 2014), Cruz-Roa *et. al.* introduced our **IDC** dataset of interest containing 277,524 patches of IDC whole slide images with 28.39% of positive samples. They developed a custom 3-layer CNN architecture which resulted in F1-score and balanced accuracy of 71.80% and 84.23% respectively. In (Janowczyk and Madabhushi, 2016), Janowczyk and Madabhushi used AlexNet architecture with additional cropping and rotations to achieve F1-score and balanced accuracy of 76.48% and 84.68% respectively on the IDC dataset. In (Romero *et al.*, 2019), Romero *et. al.*

leveraged the Inception architecture (Szegedy *et al.*, 2015) introduced by Szegedy C. *et. al.* and combined it with the regularization technique of batch normalization (Ioffe and Szegedy, 2015) to reduce internal covariate shift. Applying this approach to the IDC dataset, F1-score and balanced accuracy improved to 89.7% and 89% respectively. In (Narayanan *et al.*, 2019), Narayanan *et. al.* first pre-processed IDC patches using color constancy technique and then applied a custom 5-layer CNN architecture resulting in Area-Under-Curve (AUC) of 0.94 which was a benchmark for future efforts. In (Alghodhaifi *et al.*, 2019), Alghodhaifi *et. al.* tested 2 CNN models on the IDC dataset. One was a depth-wise separable convolution network while the other was a standard convolution network. Each of these models was tested with different activation functions – ReLU, Sigmoid, and Tanh. Standard convolution network with ReLU activation delivered the best results with 76% F1-Score, 87.13% accuracy, and 93.44% sensitivity.

Besides the BreakHis, BACH, CAMELYON and IDC datasets, there has been some interesting research on similar classification tasks in other datasets and in different domains. In (Hameed *et al.*, 2020), Hameed *et. al.* performed ensemble deep learning on their collected dataset using VGG16 and VGG19 models. The ensemble of fine-tuned VGG16 and VGG19 models delivered promising results with a sensitivity of 97.73%, an accuracy of 95.29%, and F1-score of 95.29% for the carcinoma class. In (Zhou *et al.*, 2020), Zhou *et. al.* has presented a detailed review of the classical and deep neural network techniques presented since 2012 for breast cancer histopathology image analysis. To sum up, most of the better-performing algorithms have leveraged transfer learning and CNNs to come up with innovative combinations to solve the histopathology image classification problem.

The main aim of this research is to propose a model to predict the occurrence of Invasive Ductal Carcinoma (IDC) cells in histopathology slides prepared for breast cancer detection. We seek to explore state-of-the-art transfer learning strategies and compare the performance of such architectures with that of a personalized prediction model using CNNs in order to identify a good solution which can support the pathologist in solving the IDC detection problem.

2. RESEARCH METHODOLOGY

2.1 Dataset Selection

The original dataset (Cruz-Roa *et al.*, 2014) consists of digitized whole slide images. These images were obtained from breast cancer histopathology

slides of 279 patients diagnosed with IDC at the Hospital of the University of Pennsylvania and The Cancer Institute of New Jersey. The patch-based dataset was introduced by Cruz-Roa *et. al.* (Cruz-Roa *et al.*, 2014) for IDC image classification. Only 28.4% of the 277,524 patches are IDC positive.

2.2 Dataset Preparation

A whole-slide scanner has been used for digitizing the images at 40x magnification (0.25 µm/pixel resolution). Since these images were too large for analysis, they were down sampled (by a factor of 16:1) to 4 ìm/pixel.

Using grid sampling, each whole slide image was thus divided into non-overlapping image patches. The patches exhibiting fatty tissue or slide background were discarded. Patches containing IDC were manually annotated by a pathologist using a binary annotation mask. To establish the gold standard or ground truth for training purposes, patches were labelled as positive (*i.e.* '1') if the mask covers a minimum of 80% of the patch region, otherwise, they were labelled as negative (*i.e.* '0'). This resulted in 277,524 RGB patches of 50 × 50 pixel size.

2.2.1 Data Pre-processing

The data will be randomly split into 3 subsets–training for building the initial model, validation for tuning, and testing for final evaluation. One of the problems in whole-slide image patches is that there could be large number of patches without relevant information for the classification problem. However, given that there are close to 80,000 patches which are IDC positive, this should not be a major issue in this case.

2.2.2 Data Augmentation

While the analysis of an imbalanced dataset can be managed by under-sampling, over-sampling, or algorithmic methods (Vluymans, 2019), there is a need to evaluate whether the problem of balancing the data at this stage can be addressed by data augmentation or by using a weighted binary entropy as a loss function. In order to assess the impact of data augmentation on model performance, random translations (shifts in height or width), rotations, zoom, horizontal and vertical flips will be used.

2.3 Modelling Approaches

Our custom model is a CNN-based architecture trained using labelled patch data. This architecture involves building a convolutional neural network where the convolution operation is applied on pixel information of large

images by smaller filters. This helps in extracting lower level features that are used as predictors for classification task. Once the convolution operations have been applied through a set of smaller size filters (usually one for each feature map) in a convolution layer, the data is passed through three layers:

- **Pooling layer:** This layer helps in reducing dimension of large image representation by sampling subsets of feature maps through pooling functions in a way that local invariant feature information is not lost.
- **Fully Connected layer:** This layer is usually a part of the top layer of the CNN which takes inputs from the pooling layer and converts the high-level feature data into a feature vector.
- **Classification layer:** This is the final layer and is also a fully connected layer. The number of neurons in this layer is equal to the number of classes being used for the classification task and the activation function is usually the soft-max function.

Further, these layers are also interspersed with dropout and batch normalization layers to improve the generalizability of learning and reduce over-fitting of data.

The idea is to perform experiments involving CNNs through different combinations of state-of-the-art network architectures by adopting transfer learning techniques (Tammina, 2019). Experimentation has been done with models where the pre-trained CNNs are followed by fully connected layers on top to optimize the model performance. The output of such models is compared with a custom-built CNN-based architecture involving different feature extraction methods similar to convolutional auto-encoders (Maggipinto *et al.*, 2018) along with hyperparameter tuning and optimization techniques.

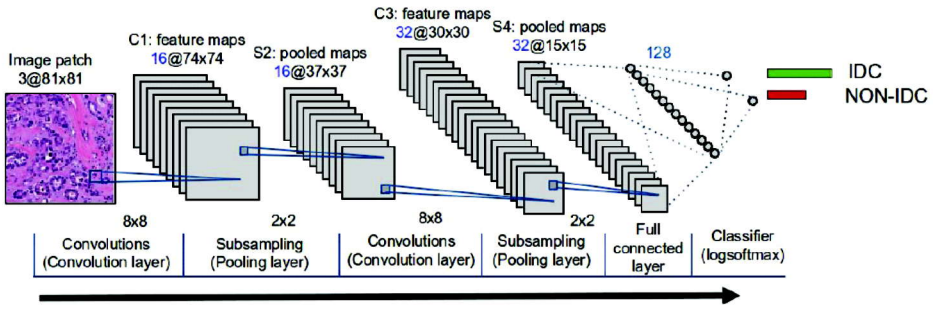

Figure 1: Sample CNN Architecture (Cruz-Roa *et al.*, 2014)

2.3.1 Transfer Learning Methods–Model Architecture

In (Weiss *et al.*, 2016), a survey of transfer learning is presented along with formal definitions. The central idea is to improve performance in a function by borrowing information from another related function. While applying deep learning techniques to classification problems, transfer learning from state-of-the-art algorithms has been one of the most common approaches. This is mainly done in two ways:

- fine-tuning predefined parameters of a state-of-the-art network
- utilizing the pre-trained network layers for feature extraction and replacing the final layers of the pre-trained network with the target classification network

In (Canziani *et al.*, 2016), the performance of most of the common state-of-the-art algorithms have been analysed in terms of accuracy, computational resources, operations, inferences, and parameter size. This helps in identifying the likely models we should focus on for transfer learning methods.

As per Figure 2, ResNet-50 provides one of the best accuracies without being very operationally intensive. ResNet-50 architecture was introduced in (He *et al.*, 2016).

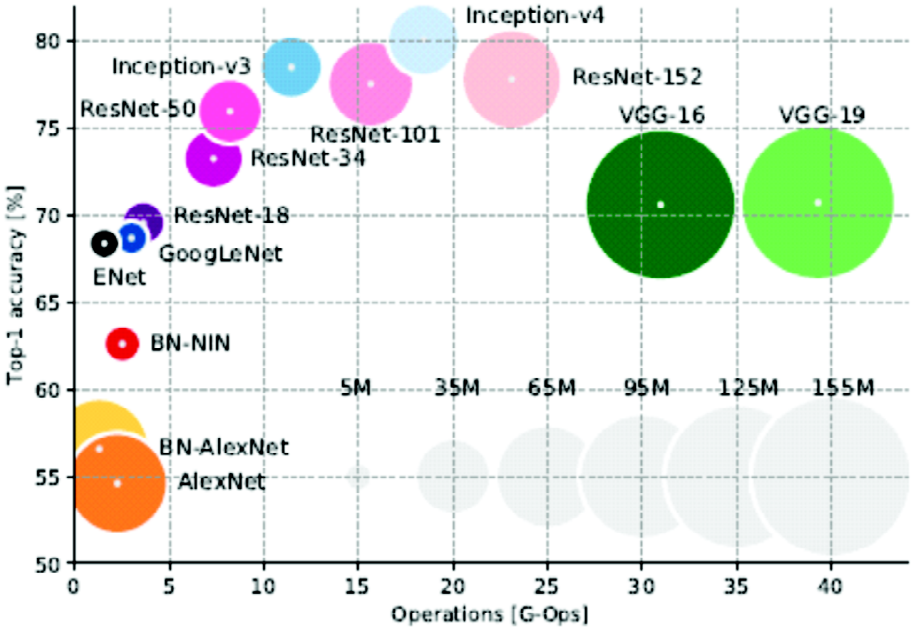

Figure 2: Top-1 Accuracy vs Operations (Canziani *et al.*, 2016)

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 3: ResNet Architecture

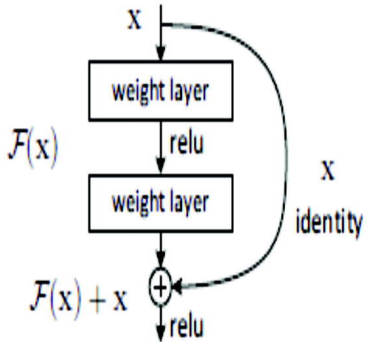

Figure 4: Residual learning

In the ResNet model, the layers are learning residual functions of the inputs coming from previous layers. This helps in preventing the performance of higher layers to be worse than the lower layers. This is achieved by inserting shortcut connections to a VGG-19 type of network.

Since ResNet-50 has been one of the most widely followed transfer learning method and has stood the test of time, we are using it as one of the state-of-the-art networks for transfer learning.

Considering the trade-off between speed and accuracy while training deep learning models, one of the most popular set of models in recent years has been the MobileNet architecture. In (Howard *et al.*, 2017), MobileNets are described as a set of efficient models since they use depth-wise separable convolution networks instead of standard convolution networks.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Table 2. Resource Per Layer Type

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

Figure 5: MobileNet Architecture

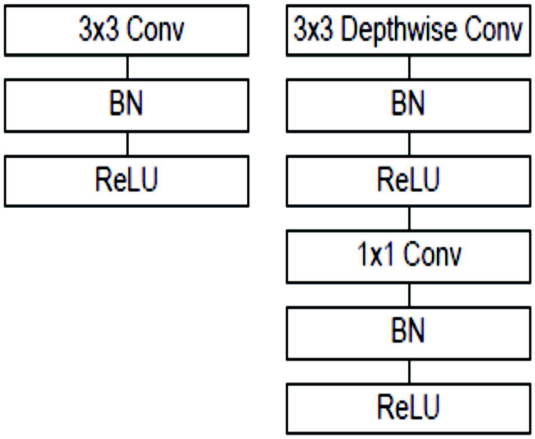

Figure 6: Standard CNNs vs Depth-wise CNNs

The MobileNet model introduces 2 global hyperparameters called width multiplier and resolution multiplier which can be changed to make the network trade off acceptable accuracy in order to improve on size and speed. This is how they achieve higher efficiencies. Since MobileNet models provide the power to deploy models on mobile and edge platforms, they can be useful in building scalable systems for breast cancer detection. So, we will be considering it for our experimentation with transfer learning.

In terms of improving accuracy and efficiency, one of the most recent architectures that have gained prominence are the EfficientNet models. In (Tan *et al.*, 2020), the concept of model scaling has been introduced which helps in balancing network depth, width, and resolution to achieve better performance. This model scaling is achieved through a compound scaling method. With the help of a compound coefficient, network depth, width and resolution are uniformly scaled in a principle way. In order to build the EfficientNet, a baseline network is initially built using a neural architecture search which optimizes accuracy and floating-point operations per second. This is then scaled up using the compound scaling method to create the final architecture. Given its accuracy and efficiency, we are also experimenting using the EfficientNet-B7 model for transfer learning.

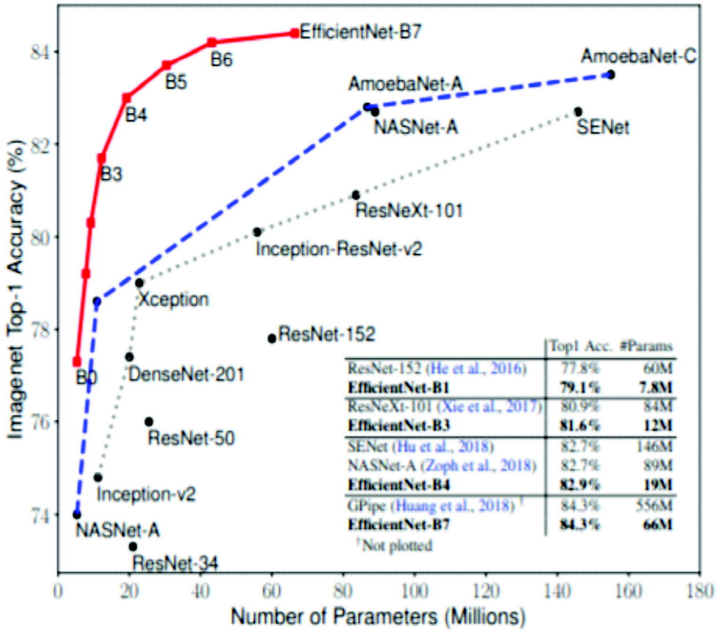

Figure 7: Top-1 Accuracy vs Number of Parameters (Tan *et al.*, 2020)

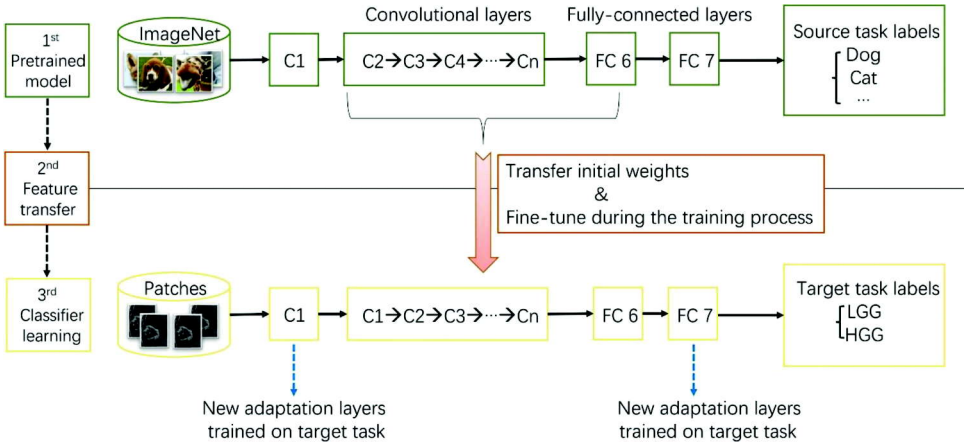

Figure 8: A typical CNN transfer learning pipeline. LGG and HGG stand for the binary classification labels of '0' and '1'.

2.3.2 *Comparative Analysis of State-of-the-Art Architectures*

ResNet50	*MobileNet-V2*	*EfficientNet- B7*	*CancerNet-SCa*
• Has several variants - ResNet-18, ResNet-34, ResNet-50, ResNet-101, ResNet-152, etc. • Convolution block, Identity block and Residual block. • Learns residual functions from inputs of previous layers. • Identity block helps address vanishing gradient problems. • Assembly of smaller networks. • Bottleneck residual block design for ResNet50 onwards-3 layers (1 × 1, 3 × 3, 1 × 1 convolutions). are stacked one over the other for each residual function. • Batch normalization helps reduce covariate shift. • One of the most popularly used CNN methods for image recognition.	• Uses depth-wise separable convolution as its basic unit which factorizes a standard convolution into a depth-wise convolution and 1 × 1 pointwise convolution. • Factorization drastically reduces computations and model size. • Based on an Inverted Residual Block structure where the residual connections are for bottleneck layers [Traditional Residual block follows a wide → narrow → wide structure but inverted residuals follows a narrow → wide → narrow approach] • Used primarily for mobile devices and embedded mobile applications.	• Used for scaling all the dimensions of images with a stable coefficient which get added to the baseline network. • Scaling helps in balancing network depth, width and resolution to achieve better performance. • Uses a compound scaling method which uses a compound coefficient to uniformly scale all dimensions of network depth, width and resolution. • Useful for using deep learning on the edge and mobile devices, as it reduces compute cost, battery usage, training and inference speeds.	• Self-attention architecture design with attention condensers. • Highly diverse and heterogenous network architecture with a mix of spatial convolutions, point-wise convolutions, depth-wise convolutions. • Number of channels in each visual attention condenser represents the number of channels for the down-mixing layer, the embedding structure and the up-mixing layer respectively • Used for detection of skin cancer from dermoscopy images.

2.3.3 Building Customized Model–Model Architecture

We are comparing the outcome of our transfer learning models with a customized CNN model which uses an architecture inspired from convolutional autoencoders for feature extraction with dropout and batch normalization after every block to improve generalizability and reduce overfitting.

Autoencoder models have two parts–Encoder and Decoder. The encoder part learns the information from the input and reduces higher dimension data to a lower dimension representation. The decoder part then uses the encoder output to regenerate the input in an unsupervised way. Once this is complete, the encoder output provides the features information which can then be used for supervised classification through the final layer. Since the autoencoders are so powerful in extracting features in lower dimensions, we expect to build an efficient network using a similar architecture.

Since Adam optimizer works better on sparse gradients and is more robust to hyperparameter changes, we will be using Adam optimizer as the optimization algorithm. In (Smith, 2017), it is shown that varying learning rates cyclically between boundary values provide better optimization in performance. So, we will be first estimating boundary values by linearly increasing learning rates for a few epochs and then varying the learning rates within these values for hyperparameter optimization.

2.3.4 Evaluation & Interpretation

The model performance will be evaluated based on standard measures of classification quality. Key metrics like Accuracy, Precision, Sensitivity (Recall), Specificity, F1-Score, G-mean, and MCC (Matthews Correlation Coefficient) will be used for comparing the model performance across experiments. In addition to this, Classification Report will be used to evaluate and visualize the health of the model's classification performance.

3. ANALYSIS AND RESULTS

3.1 Dataset Description

The original dataset (Cruz-Roa *et al.*, 2014) consisting of 277,524 patches is available as a compressed file named "IDC_regular_ps50_idx5.zip" of size 1.6 GB. Since this contains data for 279 cancer patients, each patient has a

patient ID. For each such patient ID, we have patches for both IDC positive and negative instances.

The format of the file name on each path is as follows: nxXyYclassC.png → example: 99999idx7x1521y1011class0.png. Here, n represents the first 9 characters (99999idx7) which denote the patient ID. The x-coordinate and y-coordinate of the image representing the start of the patch are denoted as X and Y respectively. The class label is given by C, *i.e.* 1 for IDC and 0 for non-IDC.

3.2 Exploratory Data Analysis

Data visualizations have been created to help us understand the distributions of patches across patients and the distributions of IDC positive and negative cases across these patches.

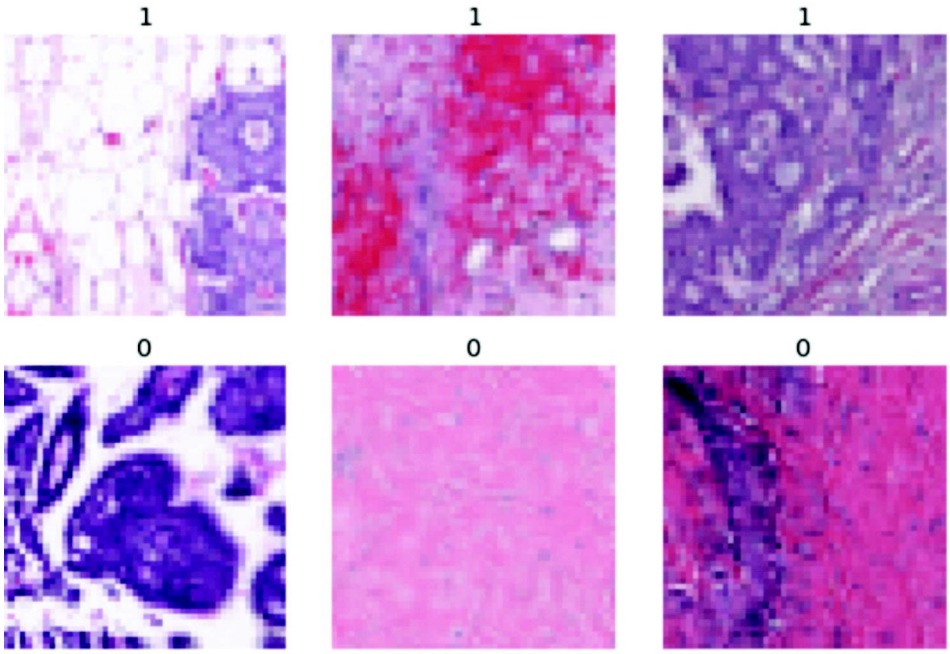

Figure 9: IDC Images. First row images having label '1' denote IDC positive patches and second row images having label '0' denote IDC negative patches.

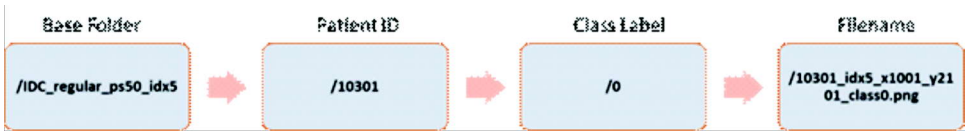

Figure 10: File directory structure of patch image dataset.

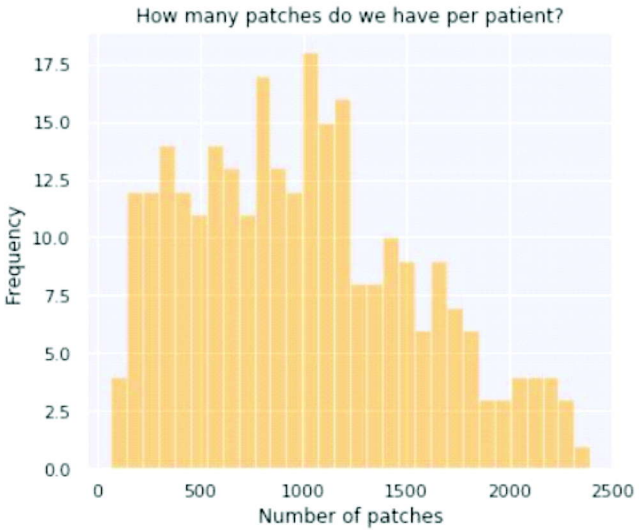

Figure 11: Distribution of number of patches across patients

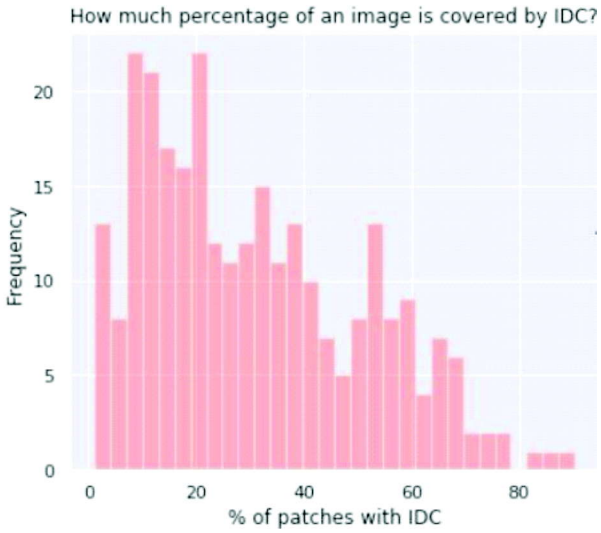

Figure 12: Percentage distribution of IDC instances across patients

In Figure 11, it is observed that the number of patches per patient varies over a wide range, as some patients have only 100+ patches and some have as much as 2400+ patches. This could mean that the resolution of cells might be changing across patients.

Figure 12 presents the proportion of IDC positive instances in the patches for each patient. While some patches have very less percentages of

IDC positive cases, there are patients who have 80% patches as IDC positive. Possible reasons for the latter cases could be that either the cancerous cells have spread across the cell patches or the region of interest predominantly focuses on the cancerous region.

Figure 13 highlights that close to 80,000 IDC positive instances contribute to only around 29% of the overall total of 277,524 patches.

Figures 14 and 15 give a quick glimpse into some of the IDC positive and negative patches. Upon comparing both these views, it is noticed that the cancerous patches tend to be more violet in colour than the non-cancerous ones. The pathologist can help us understand whether this is due to specific response of cancerous cells to test stains or due to presence of specific cells and tissues in certain patches.

Since co-ordinate information is available in the filenames of the patch images, the IDC label information on the x-y coordinates has been plotted in the form of blue areas for label '0' and red areas for label '1'. In Figure 16, it is observed that the cancer patches are commonly present in clusters. However, there are tissue patches with no information as well that is represented by the blank cells. The pathologist can help us understand if these tissues have been discarded as per testing process or these are instances of lost information.

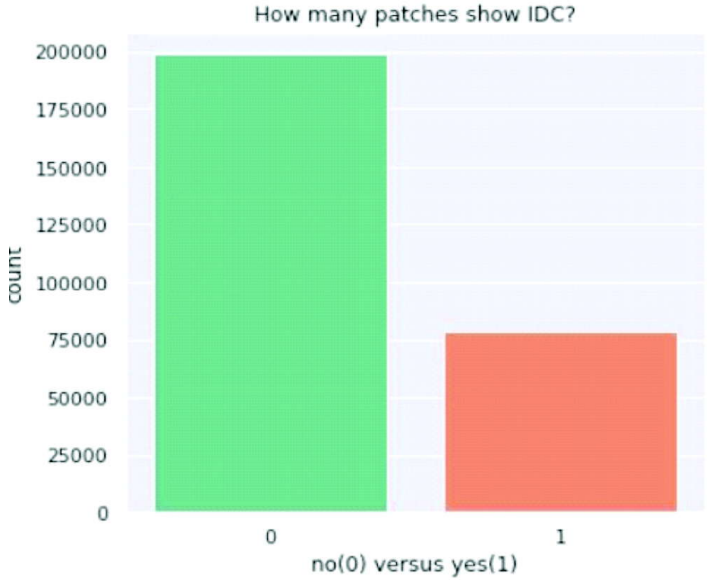

Figure 13: Number of patches showing IDC positive vs negative

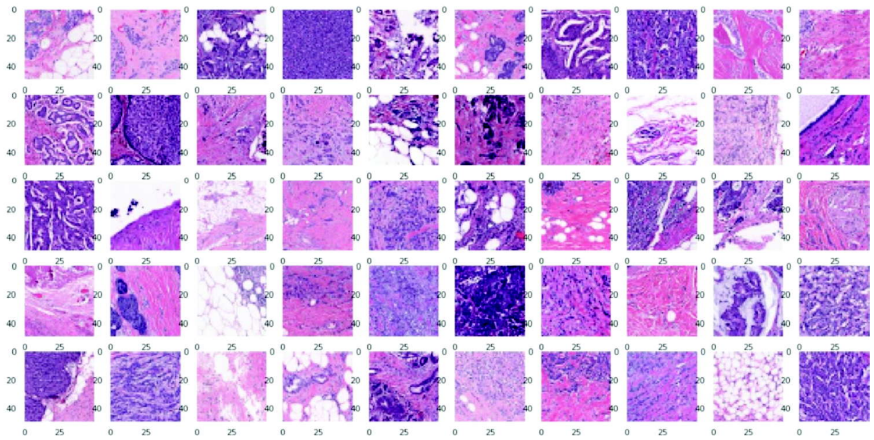

Figure 14: Cancerous patches showing IDC positive

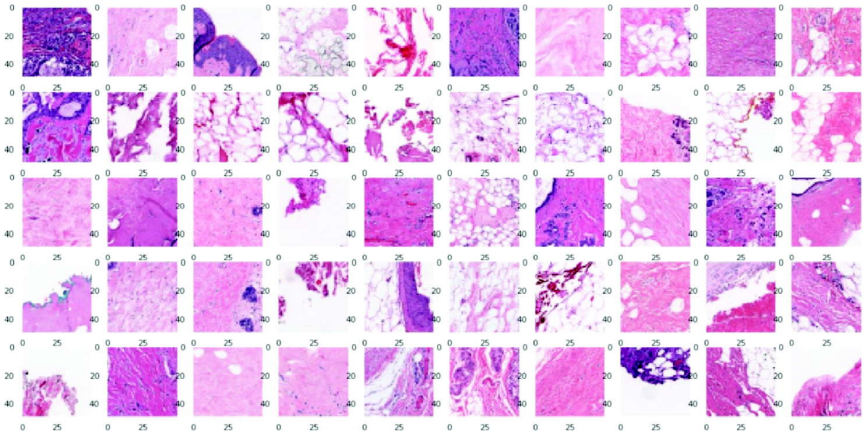

Figure 15: Non-Cancerous patches showing IDC negative

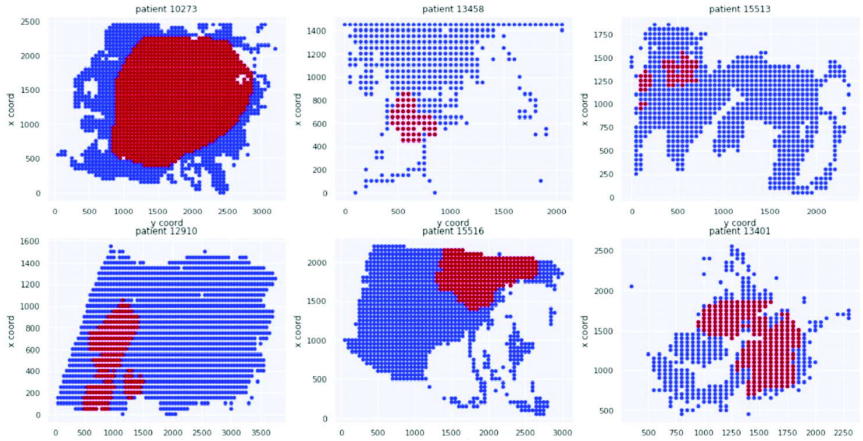

Figure 16: Representing IDC label information on patch $x - y$ coordinates

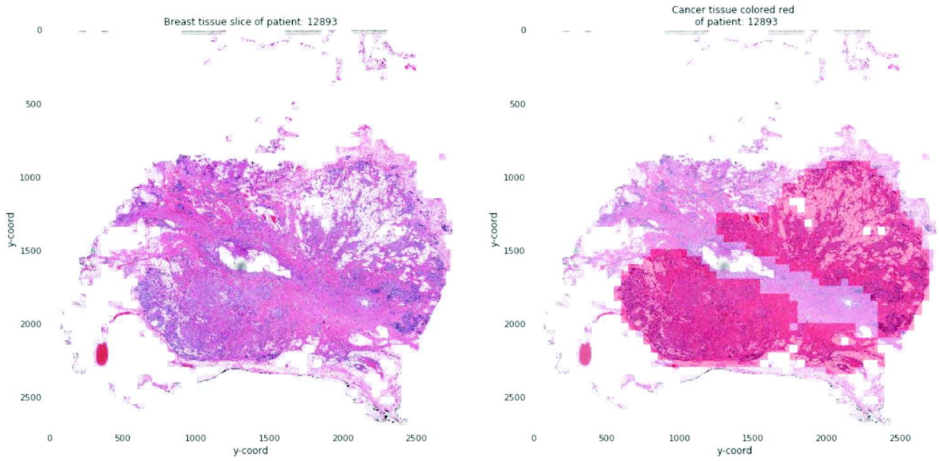

Figure 17: Overlay of IDC positive image on breast tissue for patient ID 12893

In Figure 17, the patch images have been stitched together into the entire tissue based on x-y coordinate information and the IDC positive image has been overlaid on this for a particular patient.

3.2.1 *Data Split for Training, Validation & Testing*

Before moving to data transformation and augmentation, the dataset of 277,524 images has been randomly split into 80% for training and 20% for testing purposes. 10% of the training set has been further split as validation data for tuning purposes. While data transformation techniques for target pixel size and rescaling have been applied to both training and validation data, data augmentation techniques have only been applied to training data.

3.2.2 *Data Transformation*

Since convolutional neural network architectures with pooling layers of size 2×2 have been used, pixel sizes with higher powers of 2 are preferable for analysis. Pixel size of 48×48 is very close to the original pixel size and is also equivalent to $2^4 \times 3$. So, a target size of 48x48 has been adopted for analysis of images.

Further, the images have been rescaled by dividing each pixel value by 255 so that all pixel values fall within the range of 0 to 1. The images have been rescaled by normalizing them. Only the training set of images has also been randomly shuffled to prevent memorization during the learning process. Common data augmentation techniques including random

translations through shifts in height and width, rotations, zoom, horizontal and vertical flips have been used.

3.3 Model Building

The performance of the customised model has been compared with that of 4 transfer learning models built using some of the most well-known state-of-the-art models commonly applied in other fields. These models include ResNet-50, MobileNet-V2, and EfficientNet-B7. In addition to this, there are attempts to improve upon the CancerNet algorithm proposed by Adrian Rosebrock at www.pyimagesearch.com/2019/02/18/breast-cancer-classification-with-keras-and-deep-learning/ which has been tested on this dataset earlier.

3.3.1 Customized Model

A custom CNN-based architecture that uses an encoder architecture with dropout and batch normalization after every block to improve generalizability and reduce overfitting has been built. This architecture is named as Custom CNet. It strives to reduce dimensionality by 8 times from 48 to 6 pixels during the feature extraction process.

During hyperparameter tuning, learning rate parameters (0.01, 0.005) and batch size parameters (32, 64) have been tested. The best hyperparameters were learning rate of 0.01 and batch size of 64. Adam optimizer has been used as the optimization algorithm. Binary cross-entropy has been used as the loss function. Given the significant time involved in training the model for every iteration, more hyperparameters have not been tested.

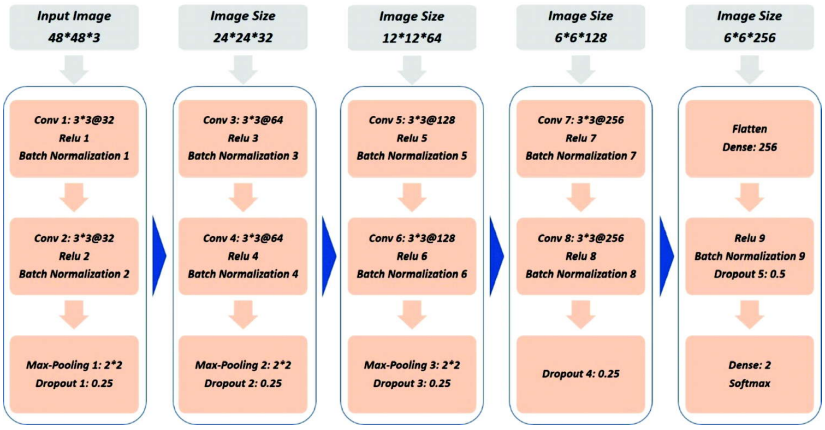

Figure 18: CustomCNet: CNN-Model Architecture

The performance of this model has been compared with that of CancerNet algorithm on the same dataset. Two versions of CancerNet algorithm were used. While the first one was the original version, the second one was the version post hyperparameter tuning.

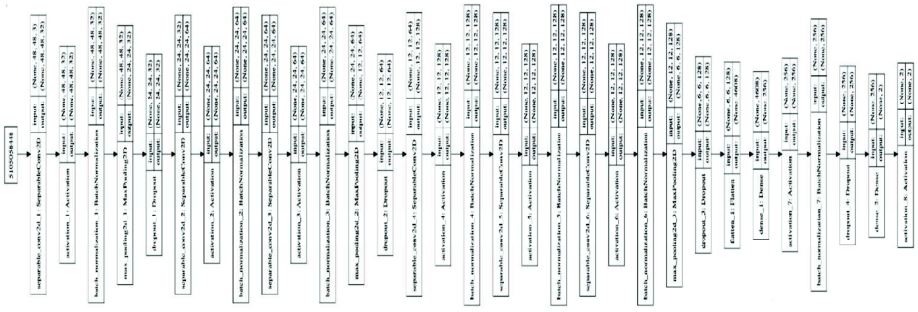

Figure 19: CancerNet Model Architecture

3.3.2 Transfer Learning

Both the common approaches used in transfer learning have been experimented:

- fine-tuning predefined parameters of a state-of-the-art network, and
- utilizing the pre-trained network layers for feature extraction and replacing the final layers of the pre-trained network with the target classification network

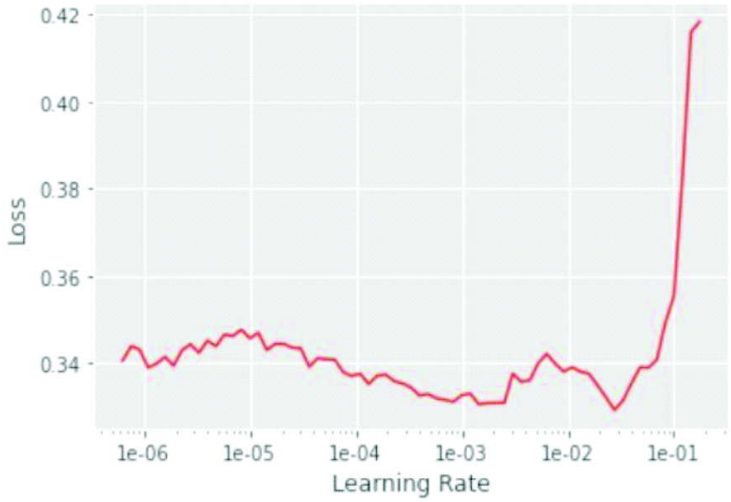

Figure 20: ResNet-50 Learning rate

Figure 21: EfficientNet_B7 Learning rate

Figure 22: MobileNet_V2 Learning rate

Among the recent literature on state-of-the-art CNN networks, MobileNet and EfficientNet architectures are being discussed a lot. This is due to their focus on efficiency as well as accuracy which makes it easy to adopt them in the practitioner world. There is not a lot of research yet on adopting some of these approaches towards diagnostic issues in healthcare. These methods have been applied in conjunction with ResNet-50 which has been one of the most common and traditional networks followed for transfer learning. The objective of the analysis is to identify the opportunity and learnings when we compare the working of the custom models with those of state-of-the-art transfer learning models.

While doing transfer learning, learning rate has been optimized by following the cyclical learning rate strategy mentioned in (Smith, 2017). By plotting the loss function against varying degree of learning rates, the region with the sharpest decline in loss function has been identified and the corresponding values of learning rate for this region have been used.

3.3.3 *Model Execution*

Keeping in mind the practical applicability of the proposed solution, the number of trainable parameters has been analysed against the time taken for training our models over 10 epochs. Since the transfer learning models have been run only on 9 epochs, the time for 10 epochs has been extrapolated for our analysis. As seen in Figure 23, while ResNet-50 and MobileNet-V2 have taken the shortest time to train, EfficientNet-B7 provides the best accuracy.

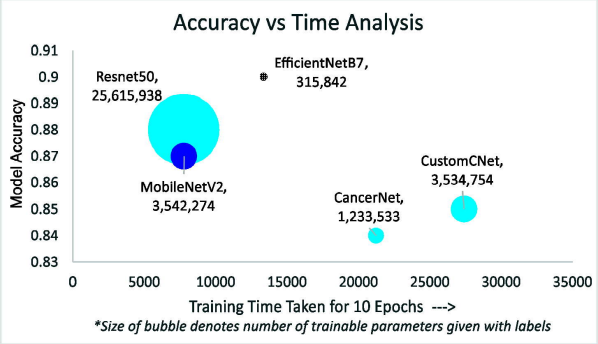

Figure 23: Model Accuracy vs Time Analysis

3.4 Model Evaluation

The model output has been evaluated through performance metrics calculated from the classification report based on model predictions. Considering that there is class imbalance in data, a range of metrics have been calculated to give us a better idea of predictive measure by effectively balancing between both false positive and false negative rates (Akosa, 2017). Most of them are derived from Sensitivity, Specificity and Precision. These metrics include Geometric Mean (G-Mean), Discriminant Power, F-Measure (F1-Score), Balanced Accuracy, Matthew's Correlation Coefficient (MCC), Youden's Index, Positive Likelihood Ratio and Negative Likelihood ratio. The objective of the model evaluation is to comment on the practicability of the solution considering the performance effectiveness in conjunction with the computational complexities.

3.5 Visualizations of Model Results

3.5.1 Custom Model–CustomCNet

```
Epoch 1/10
3122/3122 [==============================] - 3470s 1s/step - loss: 0.6366 - accuracy: 0.812
6 - val_loss: 0.5542 - val_accuracy: 0.7274
Epoch 2/10
3122/3122 [==============================] - 2628s 842ms/step - loss: 0.5157 - accuracy: 0.
8436 - val_loss: 0.4363 - val_accuracy: 0.8382
Epoch 3/10
3122/3122 [==============================] - 2650s 849ms/step - loss: 0.4895 - accuracy: 0.
8514 - val_loss: 0.3587 - val_accuracy: 0.8489
Epoch 4/10
3122/3122 [==============================] - 2677s 857ms/step - loss: 0.4744 - accuracy: 0.
8573 - val_loss: 0.3976 - val_accuracy: 0.8174
Epoch 5/10
3122/3122 [==============================] - 2670s 855ms/step - loss: 0.4565 - accuracy: 0.
8618 - val_loss: 0.4094 - val_accuracy: 0.8318
Epoch 6/10
3122/3122 [==============================] - 2645s 847ms/step - loss: 0.4431 - accuracy: 0.
8666 - val_loss: 0.4177 - val_accuracy: 0.8201
Epoch 7/10
3122/3122 [==============================] - 2656s 851ms/step - loss: 0.4356 - accuracy: 0.
8693 - val_loss: 0.3408 - val_accuracy: 0.8531
Epoch 8/10
3122/3122 [==============================] - 2657s 851ms/step - loss: 0.4313 - accuracy: 0.
8711 - val_loss: 0.3565 - val_accuracy: 0.8502
Epoch 9/10
3122/3122 [==============================] - 2668s 854ms/step - loss: 0.4192 - accuracy: 0.
8745 - val_loss: 0.3842 - val_accuracy: 0.8281
Epoch 10/10
3122/3122 [==============================] - 2682s 859ms/step - loss: 0.4203 - accuracy: 0.
8745 - val_loss: 0.3522 - val_accuracy: 0.8445
```

Figure 24: CustomCNet – Model Execution

Figure 25: CustomCNet – Model Output

Table 2: CustomCNet – Model Summary

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 48, 48, 32)        896
_________________________________________________________________
activation (Activation)      (None, 48, 48, 32)        0
_________________________________________________________________
batch_normalization (BatchNo (None, 48, 48, 32)        128
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 48, 48, 32)        9248
_________________________________________________________________
activation_1 (Activation)    (None, 48, 48, 32)        0
_________________________________________________________________
batch_normalization_1 (Batch (None, 48, 48, 32)        128
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 24, 24, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 24, 24, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 24, 24, 64)        18496
_________________________________________________________________
activation_2 (Activation)    (None, 24, 24, 64)        0
_________________________________________________________________
batch_normalization_2 (Batch (None, 24, 24, 64)        256
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 24, 24, 64)        36928
_________________________________________________________________
activation_3 (Activation)    (None, 24, 24, 64)        0
_________________________________________________________________
batch_normalization_3 (Batch (None, 24, 24, 64)        256
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 12, 12, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 12, 12, 64)        0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 12, 12, 128)       73856
_________________________________________________________________
activation_4 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_4 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 12, 12, 128)       147584
_________________________________________________________________
activation_5 (Activation)    (None, 12, 12, 128)       0
_________________________________________________________________
batch_normalization_5 (Batch (None, 12, 12, 128)       512
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 6, 6, 128)         0
_________________________________________________________________
dropout_2 (Dropout)          (None, 6, 6, 128)         0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 6, 6, 256)         295168
_________________________________________________________________
activation_6 (Activation)    (None, 6, 6, 256)         0
_________________________________________________________________
batch_normalization_6 (Batch (None, 6, 6, 256)         1024
_________________________________________________________________
conv2d_7 (Conv2D)            (None, 6, 6, 256)         590080
_________________________________________________________________
activation_7 (Activation)    (None, 6, 6, 256)         0
```

```
batch_normalization_7 (Batch (None, 6, 6, 256)         1024
dropout_3 (Dropout)          (None, 6, 6, 256)         0
flatten (Flatten)            (None, 9216)              0
dense (Dense)                (None, 256)               2359552
activation_8 (Activation)    (None, 256)               0
batch_normalization_8 (Batch (None, 256)               1024
dropout_4 (Dropout)          (None, 256)               0
dense_1 (Dense)              (None, 2)                 514
activation_9 (Activation)    (None, 2)                 0
=================================================================
Total params: 3,537,186
Trainable params: 3,534,754
Non-trainable params: 2,432
```

3.5.2 *CancerNet Model*

```
Epoch 1/10
347/347 [==============================] - 146s 419ms/step - loss: 0.7475 - acc: 0.6539
3123/3123 [==============================] - 2729s 874ms/step - loss: 0.4086 - acc: 0.8242
- val_loss: 0.7475 - val_acc: 0.6539
Epoch 2/10
347/347 [==============================] - 37s 108ms/step - loss: 0.4823 - acc: 0.7841
3123/3123 [==============================] - 2008s 643ms/step - loss: 0.3608 - acc: 0.8439
- val_loss: 0.4823 - val_acc: 0.7841
Epoch 3/10
347/347 [==============================] - 37s 107ms/step - loss: 0.3658 - acc: 0.8389
3123/3123 [==============================] - 1996s 639ms/step - loss: 0.3464 - acc: 0.8498
- val_loss: 0.3658 - val_acc: 0.8389
Epoch 4/10
347/347 [==============================] - 39s 112ms/step - loss: 0.3293 - acc: 0.8678
3123/3123 [==============================] - 2031s 650ms/step - loss: 0.3374 - acc: 0.8547
- val_loss: 0.3293 - val_acc: 0.8678
Epoch 5/10
347/347 [==============================] - 38s 109ms/step - loss: 0.3546 - acc: 0.8468
3123/3123 [==============================] - 2055s 658ms/step - loss: 0.3317 - acc: 0.8577
- val_loss: 0.3546 - val_acc: 0.8468
Epoch 6/10
347/347 [==============================] - 37s 107ms/step - loss: 0.4142 - acc: 0.8222
3123/3123 [==============================] - 2017s 646ms/step - loss: 0.3254 - acc: 0.8606
- val_loss: 0.4142 - val_acc: 0.8222
Epoch 7/10
347/347 [==============================] - 38s 109ms/step - loss: 0.4047 - acc: 0.8270
3123/3123 [==============================] - 2009s 643ms/step - loss: 0.3220 - acc: 0.8616
- val_loss: 0.4047 - val_acc: 0.8270
Epoch 8/10
347/347 [==============================] - 37s 107ms/step - loss: 0.3753 - acc: 0.8604
3123/3123 [==============================] - 2007s 643ms/step - loss: 0.3188 - acc: 0.8635
- val_loss: 0.3753 - val_acc: 0.8604
Epoch 9/10
347/347 [==============================] - 37s 106ms/step - loss: 0.3931 - acc: 0.8371
3123/3123 [==============================] - 1981s 634ms/step - loss: 0.3147 - acc: 0.8647
- val_loss: 0.3931 - val_acc: 0.8371
Epoch 10/10
347/347 [==============================] - 35s 102ms/step - loss: 0.4049 - acc: 0.8347
3123/3123 [==============================] - 1925s 616ms/step - loss: 0.3139 - acc: 0.8665
- val_loss: 0.4049 - val_acc: 0.8347
```

Figure 26: CancerNet Model Execution

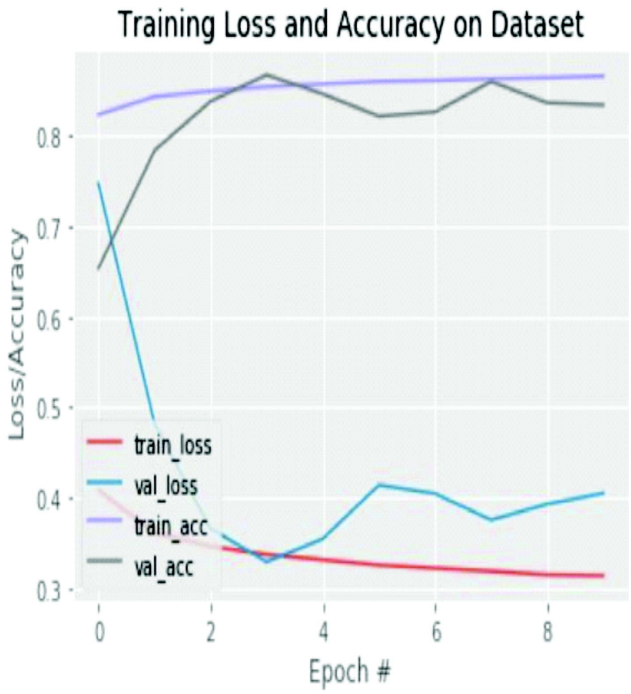

Figure 27: CancerNet Model Output

3.5.3 ResNet-50 Model

The model execution output has been shown for the two instances of transfer learning through the ResNet-50 model. Table 3 shows the output of fine-tuning the pretrained ResNet-50 model (pretrained on weights used for classification of ImageNet data). After fine-tuning the model, the layers of the model have been unfrozen and re-trained on the images using optimal learning rate for which the results are shown in Table 4.

Table 3: ResNet-50 Model Fine-Tune

epoch	train_loss	valid_loss	accuracy	time
0	0.412901	0.375586	0.834534	13:48
1	0.365206	0.336303	0.848335	13:02
2	0.333652	0.319052	0.862352	12:46
3	0.331069	0.310341	0.869469	12:46
4	0.330482	0.308528	0.867289	12:44

Table 4: ResNet-50 Model Unfreeze

epoch	train_loss	valid_loss	accuracy	time
0	0.359211	0.355639	0.842318	12:52
1	0.333413	0.317352	0.869271	12:53
2	0.306443	0.282767	0.881702	12:52
3	0.294635	0.273585	0.880279	12:52

3.5.4 MobileNet-V2 Model

The model execution output has been shown for the two instances of transfer learning through the MobileNet-V2 model. Table 5 shows the output of fine-tuning the pretrained MobileNet-V2 model (pretrained on weights used for classification of ImageNet data). After fine-tuning the model, the layers of the model have been unfrozen and re-trained on the images using optimal learning rate for which the results are shown in Table 6.

Table 5: MobileNet-V2 Model Fine-Tune

epoch	train_loss	valid_loss	accuracy	time
0	0.401807	0.374662	0.845038	13:45
1	0.357975	0.335547	0.850533	13:01
2	0.344125	0.320747	0.861740	12:46
3	0.338152	0.313371	0.865433	13:00
4	0.333811	0.312214	0.861199	12:55

Table 6: MobileNet-V2 Model Unfreeze

epoch	train_loss	valid_loss	accuracy	time
0	0.335163	0.310295	0.866280	13:00
1	0.324552	0.309694	0.862388	12:50
2	0.336208	0.307777	0.869199	12:43
3	0.324179	0.307664	0.869955	12:45

3.5.5 *EfficientNet-B7 Model*

The model execution output has been shown for the two instances of transfer learning through the EfficientNet-B7 model. Table 7 shows the output of fine-tuning the pretrained EfficientNet-B7 model (pretrained on weights used for classification of ImageNet data). After fine-tuning the model, the layers of the model have been unfrozen and re-trained on the images using optimal learning rate for which the results are shown in Table 8.

Table 7: EfficientNet-B7 Model Fine-Tune

epoch	train_loss	valid_loss	accuracy	time
0	0.346948	0.473715	0.821274	23:18
1	0.311501	0.306185	0.852299	22:54
2	0.299176	0.275587	0.879774	22:27
3	0.261325	0.261229	0.879576	22:03
4	0.251379	nan	0.898043	22:23

Table 8: EfficientNet-B7 Model Unfreeze

epoch	train_loss	valid_loss	accuracy	time
0	0.293387	0.380248	0.789997	22:12
1	0.288463	nan	0.873613	21:36
2	0.249660	nan	0.907880	21:52
3	0.248752	nan	0.901701	21:40

3.6 Evaluation of Model Performance

The performance measures for all models have been compiled in Table 9. Besides the common metrics of accuracy, sensitivity, specificity and precision, additional metrics like Geometric Mean (G-Mean), Discriminant Power, F-Measure (F1-Score), Balanced Accuracy, Matthew's Correlation Coefficient (MCC), Youden's Index, Positive Likelihood Ratio and Negative Likelihood ratio have been reported. In cases of class imbalance in data, these metrics give a better idea of how good a predictive classifier is (Akosa, 2017).

Table 9: All-Model Comparison of Performance Metrics

Models Implemented	Formula	CustomCNet	Tuned-CancerNet	Resnet50	MobileNetV2	EfficientNetB7
Accuracy	(TP + TN) / (TP + TN + FP + FN)	0.85	0.84	0.88	0.87	**0.90**
Misclassification Rate (1 - Accuracy)	(FP + FN) / (TP + TN + FP + FN)	0.15	0.16	0.12	0.13	**0.10**
Recall (or Sensitivity)	TP / (TP + FN)	**0.91**	0.89	**0.91**	0.87	**0.91**
Specificity	TN / (TN + FP)	0.82	0.82	0.87	0.87	**0.90**
Precision (or Positive Predictive Value)	TP / (TP + FP)	0.67	0.66	0.73	0.73	**0.78**
G-mean	SQRT(Sensitivity X Specificity)	0.86	0.85	0.89	0.87	**0.90**
Discriminant Power	SQRT(3)/(22/7) X (log(Sensitivity/(1-Sensitivity)) + log(Specificity/(1-Specificity))	0.92	0.86	1.01	0.91	**1.08**
F-Measure (or F1-Score)	2 X (Sensitivity X Precision) / (Sensitivity + Precision)	0.77	0.76	0.81	0.79	**0.84**
Balanced Accuracy	(Sensitivity + Specificity)/2	0.87	0.86	0.89	0.87	**0.91**
Matthew's Correlation Coefficient (MCC)	[(TP X TN) - (FP X FN)] / SQRT[(TP + FP) X (TP + FN) X (TN + FP) X (TN + FN)]	0.68	0.66	0.74	0.70	**0.78**
Youden's Index	Sensitivity - (1 - Specificity)	0.73	0.71	0.78	0.74	**0.81**
Positive Likelihood Ratio	Sensitivity / (1 - Specificity)	5.06	4.94	7.00	6.69	**9.10**
Negative Likelihood Ratio	(1 - Sensitivity) / Specificity	0.11	0.13	**0.10**	0.15	**0.10**
Time Taken		7:36:43	5:53:59	2:09:33	2:09:44	3:42:42
Trainable Parameters		3,534,754	1,233,533	25,615,938	3,542,274	315,842

As observed in Table 9, the custom model (*i.e.* CustomCNet) performs marginally better than the tuned CancerNet model. Considering the overall predictive performance, EfficientNet-B7 model outperforms other models on most of the performance metrics. In Figure 28, the performance effectiveness of models has been compared with the time taken to train them. This helps in understanding the relationship between accuracy and computational complexity and decide on which model should be the optimal choice as a classifier. Matthew's Correlation Coefficient (MCC) has been chosen as the performance metric for Figure 28, because it is one of the least impacted by imbalanced data (Akosa, 2017).

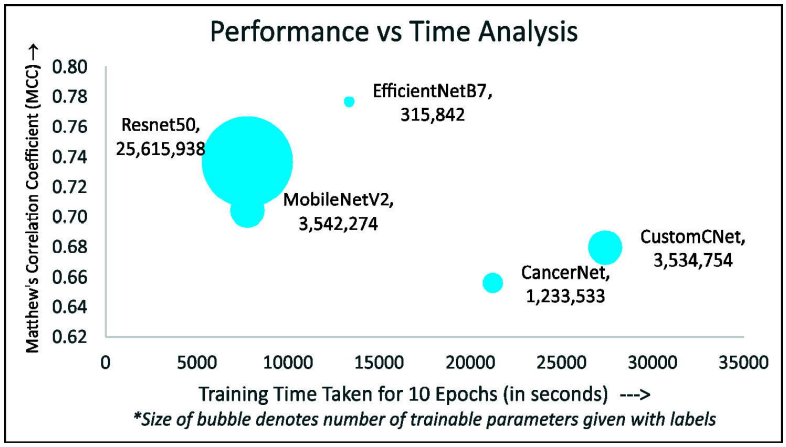

Figure 28: Model Performance – Training Time Analysis

3.7 Interpretation and Key Insights

For evaluation of classifier performance on imbalanced data, model accuracy can be misleading since the majority class will have a greater weightage to influence the results more than the minority class. In such cases, sensitivity and specificity indicate the predictive quality of minority class and majority class respectively. Since an ideal classifier should balance both sensitivity and specificity, MobileNet-V2 and EfficientNet-B7 perform better than others with EfficientNet-B7 being the best.

In case of medical diagnostics problems, false positives and false negatives are also important.

For a detection system to be effective, it should be able to identify all positive cases correctly. In cancer detection, misclassification of positive cases as negative can cost human life if the patient is not able to get detected at an early stage and treated properly. Misclassification of negative cases as positive is also undesirable as a healthy person will be putting himself through the trauma of the cancer treatment process. Hence, both recall and precision play an important role in determining the effectiveness of a medical diagnostic process. An ideal system should be able to balance both. This is where our additional performance metrics help us identify models with better predictive quality. Metrics like MCC, F1-Score and Discriminant Power do a better job in evaluating the predictive power of a classifier. In most of these measures, the CustomCNet model performs marginally better than the CancerNet model. However, the transfer learning models built on state-of-the-art networks perform much better than both these models. EfficientNet-B7 provides the best performance among all the models on most of the performance metrics.

As observed in Figure 28, the model performance comes at the cost of computational complexity with trainable parameters in the order of millions taking anything from 2+ hrs to 7+ hrs of training time. One of the reasons why transfer learning models have dominated deep learning applications for image classification is their ability to train faster with better predictive performance despite complex architectures and high number of trainable parameters. This is what has been seen in these experiments as well, especially with the ResNet-50 model. Further, with some of the recent advancements in deployment for cloud, mobile and edge platforms, efficient architectures like MobileNet-V2 and EffficientNet-B7 make great transfer learning models, especially for being deployed for practical applications in industry. Being able to build such models at low cost and

making them deployable over cloud, mobile and edge platforms will be key to developing scalable systems. Such systems can help in expanding state-of-the-art decision support and possibly improve mortality rate for breast cancer across geographies. Since these models can be extended across domains, similar systems can be built for most other medical diagnostics services as well.

Despite providing almost similar performance levels, the CustomCNet model takes longer time to train than CancerNet model. This is because CancerNet uses depth-wise separable convolution layers which are more efficient and require less computation than standard CNN layers. Depth-wise separable CNNs are also a part of the MobileNet-V2 architecture and contribute to their being lighter and more efficient.

4. CONCLUSIONS AND RECOMMENDATIONS

It is important to note that the last mile implementation of the AI solution or deep learning model will most likely be monitored or handled by the pathologist. Therefore, his role in conjunction with the deep learning systems for medical diagnostics is crucial for successful delivery. During our analysis, we realized that the deep learning prediction model can't just replace the pathologist job in detecting breast cancer. It will be most effective when it will provide its prediction to the pathologist as an intelligent decision support system with relevant measures on key performance metrics. This will help the pathologist identify specific problem areas to address. By adopting standardized prediction systems, model predictions can help pathologist make lesser mistakes which could arise due to difference in skill levels or non-availability of enough time due to overwhelming workload. Evaluating recall and precision along with focusing on false positives or false negatives arising due to conflict in model and pathologist predictions can help save more human lives. In breast cancer detection problems, it is better to err on the side of caution (*i.e.* false positive) or detect disease early enough rather than discover a false negative later at advanced stages.

Further, the pathologist can play an important role in better understanding of the patch image data. During exploratory data analysis, we found the pathologist's insights will be useful to understand whether some of the data issues are genuine or arising due to biological reasons. For example, the pathologist can help us understand whether the violet colour intensity in patches is happening due to specific response of cancerous cells to test stains or due to presence of specific cells and tissues

in certain patches. The pathologist can also help evaluate the gaps in patch image data and assess whether these are due to deliberate discarding of partial tissue or loss in information due to non-availability of patches. One of the problems in whole-slide image patches is that there could be large number of patches without relevant information for the classification problem. In (Golatkar *et al.*, 2018), patches have been selected for analysis based on nuclear density represented by higher intensity of bluish pixels. This helps us focus on patches with relevant information. Such analysis can be further developed with help of the pathologist's inputs.

We understand that the CustomCNet model delivered good results due to the encoder-like architecture which helps us in reducing dimensionality as we proceed through the feature extraction process. The process of establishing ground truth for identifying positive or negative instances in breast cancer detection and other medical diagnostics cases has dependencies on the pathologist. However, the pathologist is usually overwhelmed with a lot of workload due to increasing number of patients and non-availability of many skilled pathologists. This makes ready availability of correctly labelled data very difficult. Autoencoder architectures can help us address this issue by using unsupervised methods for initial clustering and feature extraction followed by additional neural network layers along with limited supervised data for final classification.

One of the key takeaways of our analysis has been the power of transfer learning through state-of-the-art network architectures like MobileNet-V2 and EfficientNet-B7 in terms of both classifier predictive quality as well as efficiency in computational capacity. Comparing with existing literature on the dataset we analysed for IDC detection, our best transfer learning model based on EfficientNet-B7 outperforms two of the earlier approaches employed in (Cruz-Roa *et al.*, 2014) and (Janowczyk and Madabhushi, 2016). Based on available literature of work done on our dataset so far, the comparable parameters comprise accuracy, sensitivity, F1-score and balanced accuracy. The EfficientNet-B7 based transfer learning model has achieved accuracy of 90%, sensitivity of 91%, F1-score of 84% and balanced accuracy of 91%.

Such transfer-learning based architectures hold a lot of promise in developing scalable solutions that are deployable across web, cloud, mobile and edge platforms without affecting performance and speed. This is particularly important in the field of medical diagnostics and breast cancer detection since it facilitates building systems for easier dissemination of pathological services over distant geographies.

Though existing transfer models have been developed to solve image classification in other domains, they still provide encouraging results in classification problems for disease detection as well. This is because the low-level features may not be very different across domains. These models do a good job in extracting low-level features and feed into custom fully connected and dense layers at the top. However, there are opportunities to further improve the classification performance for a lot of disease detection problems as models with high-performance measures are critical to saving human life in such cases. In order to accomplish this, there is a strong need to develop state-of-the-art networks on image classification problems specific to healthcare domain and make more relevant image datasets publicly available covering patients across demographic and biological profiles.

5. FUTURE WORK

There are opportunities to improve in terms of hyperparameter tuning using random search and grid search methods combined with more innovative CNN architectures. Since the execution run-time was ranging from 2+ hours to 7+ hours for our models, such techniques couldn't be used due to time constraints. While implementing these techniques on simple models may not surpass the performance of state-of-the-art transfer learning models, applying them on more innovative architectures like ensemble deep learning algorithms and multiple-staged CNN architectures might provide state-of-the-art outcomes.

Explainable AI methods using class activation maps can be used to study the activations across the CNN layers in more detail. However, a medical expert may be needed to understand these activations accurately since recognizing meaningful activations at a cellular level for histopathological studies is much more nuanced than studying activations of features across common image classification problems including humans, cats, dogs, flowers, etc. This is an area where deep learning and CNNs can assist histopathological research as well.

Developing state-of-the-art networks with publicly available architectures and parameters on image classification problems specific to healthcare domains will lead further advancement in leveraging more powerful transfer learning models for IDC detection and similar tasks. Having more publicly available image datasets covering patients across demographic and biological profiles will also benefit such research.

Continued focus on identifying architectures that are computationally lighter, efficient and more accurate will help in building more powerful and scalable systems that can be practically deployed across cloud, mobile and edge platforms.

References

1. Akosa, J.S., (2017). Predictive accuracy : A misleading performance measure for highly imbalanced data. SAS Global Forum, [online] 942, pp.1–12. Available at: https://support.sas.com/resources/papers/proceedings17/0942-2017.pdf.
2. Alghodhaifi, H., Alghodhaifi, A. and Alghodhaifi, M., (2019). Predicting Invasive Ductal Carcinoma in breast histology images using Convolutional Neural Network. *Proceedings of the IEEE National Aerospace Electronics Conference, NAECON*, 2019-July, pp.374–378.
3. BREASTCANCER.ORG, (2020). *Invasive Ductal Carcinoma: Diagnosis, Treatment, and More*. [online] Available at: https://www.breastcancer.org/symptoms/types/idc [Accessed 23 Sep. 2020].
4. Canziani, A., Paszke, A. and Culurciello, E., (2016). An Analysis of Deep Neural Network Models for Practical Applications. [online] pp.1–7. Available at: http://arxiv.org/abs/1605.07678.
5. Cruz-Roa, A., Basavanhally, A., González, F., Gilmore, H., Feldman, M., Ganesan, S., Shih, N., Tomaszewski, J. and Madabhushi, A., (2014). Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks. *Medical Imaging 2014: Digital Pathology*, 9041216, p.904103.
6. Golatkar, A., Anand, D. and Sethi, A., (2018). Classification of Breast Cancer Histology Using Deep Learning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10882 LNCS, pp.837–844.
7. Hameed, Z., Zahia, S., Garcia-Zapirain, B., Aguirre, J.J. and Vanegas, A.M., (2020). Breast cancer histopathology image classification using an ensemble of deep learning models. *Sensors (Switzerland)*, 2016, pp.1–17.
8. He, K., Zhang, X., Ren, S. and Sun, J., (2016). Deep residual learning for image recognition. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December, pp.770–778.
9. Hu, J., Shen, L., Albanie, S., Sun, G. and Wu, E., (2020). Squeeze-and-Excitation Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 428, pp.2011–2023.
10. Ioffe, S. and Szegedy, C., (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *32nd International Conference on Machine Learning, ICML 2015*, 1, pp.448–456.
11. Janowczyk, A. and Madabhushi, A., (2016). Deep learning for digital pathology image analysis: A comprehensive tutorial with selected use cases. *Journal of Pathology Informatics*, 71.

12. Jiang, Y., Chen, L., Zhang, H. and Xiao, X., (2019). Breast cancer histopathological image classification using convolutional neural networks with small SE-ResNet module. *PLoS ONE*, 143, pp.1–21.
13. Li, X., Radulovic, M., Kanjer, K. and Plataniotis, K.N., (2019). Discriminative Pattern Mining for Breast Cancer Histopathology Image Classification via Fully Convolutional Autoencoder. *IEEE Access*, 7, pp.36433–36445.
14. Maggipinto, M., Masiero, C., Beghi, A. and Susto, G.A., (2018). A Convolutional Autoencoder Approach for Feature Extraction in Virtual Metrology. *Procedia Manufacturing*, [online] 17, pp.126–133. Available at: https://doi.org/10.1016/j.promfg.2018.10.023.
15. Narayanan, B.N., Krishnaraja, V. and Ali, R., (2019). Convolutional Neural Network for Classification of Histopathology Images for Breast Cancer Detection. *Proceedings of the IEEE National Aerospace Electronics Conference, NAECON*, 2019-July, pp.291-295.
16. Nuruddin Qaisar Bhuiyan, M., Shamsujjoha, M., Ripon, S.H., Proma, F.H. and Khan, F., (2019). *Transfer Learning and Supervised Classifier Based Prediction Model for Breast Cancer*. [online] *Big Data Analytics for Intelligent Healthcare Management*. Elsevier Inc. Available at: http://dx.doi.org/10.1016/B978-0-12-818146-1.00004-0.
17. Romero, F.P., Tang, A. and Kadoury, S., (2019). Multi-level batch normalization in deep networks for invasive ductal carcinoma cell discrimination in histopathology images. *Proceedings - International Symposium on Biomedical Imaging*, 2019-April, pp.1092–1095.
18. Saxena, S., Shukla, S. and Gyanchandani, M., (2020). Breast cancer histopathology image classification using kernelized weighted extreme learning machine. *International Journal of Imaging Systems and Technology*, October 2019, pp.1–12.
19. Simonyan, K. and Zisserman, A., (2015). Very deep convolutional networks for large-scale image recognition. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, pp.1–14.
20. Smith, L.N., (2017). Cyclical learning rates for training neural networks. *Proceedings - 2017 IEEE Winter Conference on Applications of Computer Vision, WACV 2017*, April, pp.464–472.
21. Spanhol, F.A., Oliveira, L.S., Petitjean, C. and Heutte, L., (2016). A Dataset for Breast Cancer Histopathological Image Classification. *IEEE Transactions on Biomedical Engineering*, 637, pp.1455–1462.
22. Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., (2015). Going deeper with convolutions. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 07-12-June, pp.1–9.
23. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z., (2016). Rethinking the Inception Architecture for Computer Vision. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016-December, pp.2818–2826.
24. Tammina, S., (2019). Transfer learning using VGG-16 with Deep Convolutional Neural Network for Classifying Images. *International Journal of Scientific and Research Publications (IJSRP)*, 910, pp 9420.

25. Vluymans, S., (2019). Learning from imbalanced data. *Studies in Computational Intelligence*, 8079, pp.81–110.
26. Weiss, K., Khoshgoftaar, T.M. and Wang, D.D., (2016). *A survey of transfer learning*. *Journal of Big Data*, Springer International Publishing.
27. WHO, (2020). *WHO | Breast cancer*. [online] Available at: https://www.who.int/cancer/prevention/diagnosis-screening/breast-cancer/en/# [Accessed 23 Sep. 2020].
28. WHO, (2020). *Who Report on Cancer*: setting priorities, investing wisely and providing care for all. Geneva: World Health Organization; 2020. Licence: CC BY-NC-SA 3.0 IGO
29. Xie, J., Liu, R., Luttrell, J. and Zhang, C., (2019). Deep learning based analysis of histopathological images of breast cancer. *Frontiers in Genetics*, 10FEB, pp.1–19.
30. Xu, B., Liu, J., Hou, X., Liu, B., Garibaldi, J., Ellis, I.O., Green, A., Shen, L. and Qiu, G., (2019). Look, investigate, and classify: A deep hybrid attention method for breast cancer classification. *Proceedings - International Symposium on Biomedical Imaging*, 2019-April, pp.914–918.
31. Yan, R., Ren, F., Wang, Z., Wang, L., Zhang, T., Liu, Y., Rao, X., Zheng, C. and Zhang, F., (2020). Breast cancer histopathological image classification using a hybrid deep neural network. *Methods*, [online] 173June, pp.52–60. Available at: https://doi.org/10.1016/j.ymeth.2019.06.014.
32. Zhou, X., Li, C., Rahaman, M.M., Yao, Y., Ai, S., Sun, C., Wang, Q., Zhang, Y., Li, M., Li, X., Jiang, T., Xue, D., Qi, S. and Teng, Y., (2020). A Comprehensive Review for Breast Histopathology Image Analysis Using Classical and Deep Neural Networks. *IEEE Access*, 8, pp.90931–90956.

List of Abbreviations

IDC…..…Invasive Ductal Carcinoma

CNN…..…Convolutional Neural Network

GPU…..…Graphical Processing Unit

POMDP…Partially Observed Markov Decision Process

RNN…….Recurrent Neural Network

LSTM…...Long Short-Term Memory

PCA…..…Principal Component Analysis

SVM…....Support Vector Machines

Requirements and Resources

- Data: IDC Breast Cancer Histopathology dataset: This dataset was obtained from breast cancer histopathology slides of 162 patients diagnosed with IDC at the Hospital of the University of Pennsylvania and The Cancer Institute of New Jersey.

a. URL: http://andrewjanowczyk.com/wp-static/IDC_regular_ps50_idx5.zip

- Technology Stack: 'Table 1: Selection of technology stack'

#	*Area*	*Description*
1.	Programming Language	Python 3.x will be used for coding
2.	Development Environment	Jupyter notebook will be used as a development environment
3.	Data Format	In most cases, data will be stored in the file system as 'png' and will be loaded into memory with the numpy library as a numpy array(s). Also, there will be a few more data formats to store and load data between the file system and memory.
4.	Python library for CNN	There are many open-source packages available in Python for working with CNN-based models. Packages 'torch' and 'fastai' provide a lot of features to train deep learning models. Also, packages 'Keras' along with 'Tensorflow' gives the user a lot of control on designing the model. All these packages will be tried for this research since there could be advantages with each one for different levels of computationally intensive tasks.
5.	Pre-Processing, Feature Extraction	Standard python libraries will be used for both Pre-processing and Feature Extraction.
6.	Data Pipeline	Data pipeline will consist of API based communication with data exchange formats on both disk and memory.
7.	Scalability	Cloud-based scalable processing technology stack, database, data model, and architecture will be validated and recommended at the end of the research.