Comparing the Performance of Different Sized Convolutional Neural Networks on Cancer Identification

Anooshka Pendyal

2021 Summer STEM Institute

July 31, 2021

Abstract

Invasive ductal carcinoma (IDC) is the most common subtype of breast cancer, and it is important to identify it as quickly and accurately as possible. Currently, identification of IDC is done by hand, which is time-consuming and challenging. Machine learning, specifically convolutional neural networks, can be applied to assist in the diagnosis process. Currently, there is limited information about the effect of the number of layers a ResNet, a type of convolution neural network, has on the model's performance in cancer identification. In this project, the dataset was obtained from Kaggle, downsampled to address class imbalance, and preprocessed to prevent overfitting. Then, two models using different ResNets (ResNet34 and ResNet50) were trained and evaluated. One surprising result was that ResNet34 performed better during testing (F1 score of 0.71), while ResNet50 performed better during training (F1 score of 0.75) and validation (F1 score of 0.75). Generally, the deeper the neural network is, the more powerful it is, and the better it performs. Another result was that the validation accuracy and loss for both models was very unstable. This could be the result of insufficient training. Although the results of this project are inconclusive and the models do not perform well enough for real world use, there is still great potential for real world application, especially in assisting in the cancer identification process. It is important to study pre-trained models, as they reduce the amount of data needed to train a model, and data is difficult to access in healthcare.

1 Introduction

1.1 Invasive Ductal Carcinoma

According to the World Health Organization, in 2020, female breast cancer was the most commonly diagnosed cancer, with an estimated 2.3 million new cases [1]. Of the phenotypic subtypes of breast cancers, invasive ductal carcinoma (IDC) is the most common, accounting for 80 percent of subtypes [2]. IDC begins in the milk duct of the breast and spreads to surrounding fibrous and fatty tissue. Early diagnosis improves the chances of long-term survival [3]. Therefore, it is important to identify IDC as quickly and accurately as possible so that the aggressiveness of the cancer can be graded. The first step in grading the aggressiveness of a whole mount sample is to identify exact regions of IDC. Currently, identification of IDC is performed by pathologists manually scanning large areas of benign regions to identify the areas of malignancy. This process is often time consuming and challenging [2].

1.2 CNNs in Cancer Diagnosis

In recent years, there has been interest in applying machine learning to various fields of healthcare, such as cancer diagnosis. This brings many advantages including a faster diagnosis process and checking for human errors. Doctor-assisting models have also become a topic of interest in machine learning. Deep learning, a field of machine learning that focuses on neural networks, has gained attention due to success in different computer vision and pattern recognition tasks. Deep learning architectures have shown to be successful in the automated segmentation and classification of diseases [2]. A specific type of neural network, convolutional neural networks (CNNs) are well-suited to classifying images, as they are very strong in pattern recognition on 2 dimensional data, such as images and videos [4]. CNNs can be applied to slide images of breast cancer specimen to identify IDC.

ResNet is a popular pre-trained CNN model, and it comes in different sizes. Generally, the bigger the ResNet model is (the more layers it has), the more powerful it is. However, to the best of my knowledge, there has not been much research in applying ResNets of different sizes to cancer identification.

2 Literature Review

2.1 Convolutional Neural Networks

A notable strength of CNNs is that they are translation invariant. Opposed to multilayer perceptrons, CNNs view the pixels of an image in relation to nearby pixels. This means that it can look for features in different locations. Using a filter, the CNN looks for a specific feature, like a snout in pictures of dogs, and notes the number of times and in what locations the feature appears [4].

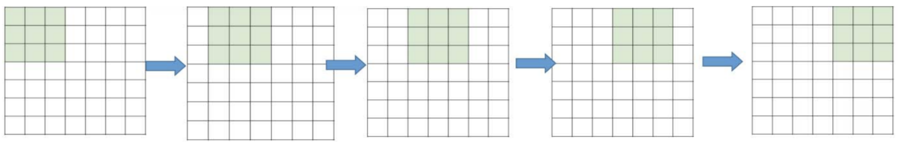

Figure 1: A diagram of how filters move across the input image.

The following 4 layers are common among CNNs: convolutional, non-linearity, pooling, and classification. In the first operation, the convolution operator extracts features from the input image by preserving the relationship between pixels. This allows it to learn image features. The purpose of the second layer, non-linearity, is to introduce non-linearity to the CNN because most real world data is non-linear. This is done by applying ReLU, which replaces all negative pixel values in the feature map with zero. The pooling, or downsampling, layer reduces the dimensions of each feature map but retains the most important information. The purpose of the previous layers was to create outputs that represent the features of the input image. The classification, or fully connected layer, uses these outputs to classify the input image into classes. For each class, this layer outputs the probability that the input image belongs to it, and the final output is the class that the input image most likely belongs to [4].

2.2 ResNet

When building models, transfer learning is often utilized. Transfer learning uses pre-trained models, which are models created by someone else to solve a similar problem. The weights, which represent what the pre-trained model has learned, are used in the user's model. Pre-trained models are used commonly, as they reduce the amount of training data need and the amount of time it takes for the model to learn. One well known pre-trained model is ResNet, which uses weights based off of the ImageNet dataset. ResNet uses skip connection to fit the input from the previous layer to the next layer without any modifications. This allows it to have deeper networks [5]. There are ResNets of different sizes with different amounts of layers. For example, ResNet50, one of the most popular ResNets, has 50 layers. ResNets come in the following sizes: 18, 34, 50, 101, and 152. Generally, the deeper the ResNet is, the more powerful it is [5]. However, to the extent of my knowledge, there has been little research done comparing the performance of different ResNet sizes in cancer identification. It is important to analyze the impact that model size has on performance, especially accuracy and training time, as it is crucial to diagnose patients accurately and efficiently.

2.3 Limitations and Purpose

The use of CNNs in classifying images of cancer is common. However, to the best of my knowledge, there are currently no studies analyzing the performance of pre-trained networks of different sizes, specifically ResNet, on classifying cancer images. For example, Alazani et al. assessed the application of CNNs on mammograms to assist in diagnosing breast cancer. They explored using CNNs of different depths, and the deepest model performed the best [6]. However, this study used CNNs developed by the authors and did not address pre-trained models. Similar studies can be found, but none of them address using pre-trained models of different sizes, specifically ResNet, on cancer classification. It is important to analyze the performance of pre-trained models in a health-care setting because pre-trained models reduce the amount of data needed, and data is difficult to get access to in healthcare. This paper addresses this gap in knowledge by aiming to compare the performance of ResNets of different sizes (34-layer and 50-layer) by using whole mount slide images of breast cancer specimens that do or do not have invasive ductal carcinoma.

3 Methodology

3.1 Data Acquisition

The data used in this project were obtained from a publicly available dataset on Kaggle called ”Breast Histopathology Images”. The dataset contains 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. In total, 198,738 patches were IDC negative, while 78,786 were IDC positive. In this project, the information regarding which patient each patch belonged to was not considered, as the models were only trained to identify whether each individual patch had IDC or not. The dataset is organized as follows: 1) a patient number corresponds with one whole mount slide image, 2) within each patient's folder, the 50 x 50 patches are created, 3) based on whether or not a patch has IDC, it is sorted into folders of 0 (IDC negative) and 1 (IDC positive). The following figure summarizes how the dataset is divided:

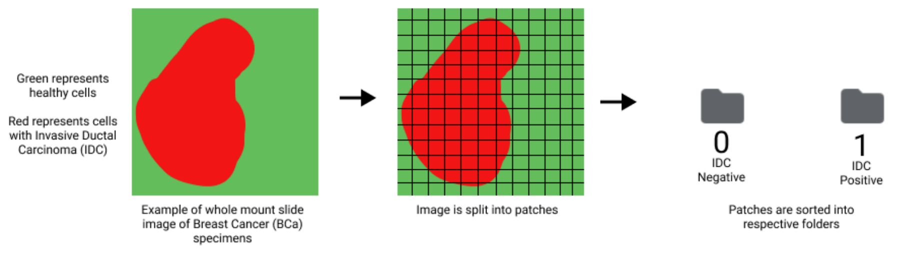
Green represents healthy cells
Red represents cells with Invasive Ductal Carcinoma (IDC)
Example of whole mount slide image of Breast Cancer (BCa) specimens
Image is split into patches
0
IDC Negative
1
IDC Positive
Patches are sorted into respective folders

Figure 2: A diagram of how the dataset is divided.

3.2 Data Downsampling

Only 50, as opposed to the complete 162, whole mount slide images were used in this project due to time constraints and computational constraints. Simply uploading the files into Google Drive took hours for each slide image, which was a clear indication that the dataset was too large for the computational resources used in this project. Then, the dataset was split into a training set, validation set, and testing set. 40 slide images were used for training, 5 slide images were used for validation, and 5 slide images were used for testing. However, in these subsets, there were class imbalances, meaning the two class (IDC negative and IDC positive) were not equally represented. For the training set, there were nearly 3 times as many IDC negative patches than IDC positive. For the validation set, this imbalance was nearly 3.5, while for the testing set, the imbalance was more than 8 times.

In order to address the issue of class imbalance, downsampling was implemented. The number of IDC positive patches was smaller in comparision to IDC negative patches, so some IDC negative patches were removed. IDC negative patches were randomly removed from the dataset until the two classes were roughly equal. This process was done for the training set, validation set, and testing set. Although downsampling is a concern in data science, it was suitable for this project, as there was a sufficient amount of data. In total, 25,975 images were used for training, 977 for validation, and 957 for testing. In each of these sets, the division of IDC negative and IDC positive was roughly even.

3.3 APIs and Libraries

The Keras API was used to create the models because it is straight forward and high-level. Keras is also commonly used in the machine learning field. ResNet34 is not included in Keras's list of available pre-trained models, so a library called "Classification Models Zoo - Keras" from the Github user "qubvel" was imported in.

3.4 Setting Model Up

First, the ResNet model was loaded in, with the weights being set to "imagenet". For each ResNet model being tested (34 and 50), the respective model was loaded in. The "include_top" parameter was set to False so that the input shape would not be changed and the weights would be compatible. Next, the existing weights were set to not be trained because the model would be using ImageNet weights. Then, the layers were flattened to become a 1-d array of elements. After that, a model object was created with the number of classes being set to two (binary classification) and the activation being set to sigmoid (used in binary classification). Finally, the loss was specified as binary crossentropy, the optimizer was specified as "adam", and the metrics were specified. "Adam" was specified as the optimizer because it is commonly used. The metrics that were specified to be output by the program were accuracy, AUC, precision, recall, true positives, true negatives, false positives, and false negatives.

3.5 Data Pre-processing

The first step in data pre-processing was to rescale the RGB values from 0-255 to 0-1. This was done because typical values of 0-255 would be too high for the model to process. This was done for all images. Then, augmentation was applied to the training data and validation data to prevent overfitting. The transformations made included shear range being set to 0.2, zoom range being set to 0.2, and horizontal flip being set to true. These transformations are applied so that the model will never see the same image twice while training.

3.6 Training and Evaluating

The file paths to the training, validation, and testing datasets were specified. Batch size, the number of samples passed through the model at a time, was specified to be 32. Then, the model was fit. Early stopping was used to determine the number of epochs to train for. Early stopping was set to monitor validation loss so that when validation loss stopped improving, the model would stop training. The patience parameter in early stopping was set to 5 so that the model would stop training once the validation loss had stopped improving for 5 epochs. Finally, after training, the model was evaluated using the test set.

3.7 PC Specifications

All programs were run on Google Colab using the NVIDIA-SMI 470.42.01 GPU. The computer used during experimentation was a MacBook Pro on the operating system macOS Big Sur Version 11.4.

4 Results and Discussion

4.1 Metrics

The metrics used for evaluation in this project were F1 score, accuracy, precision, and recall. F1 score was used as the primary measure of performance. F1 score shows the balance between precision and recall. The formula is below:

$$F_1 = 2 * \frac{Precision * Recall}{Precision + Recall}$$

Precision finds the proportion of positive identifications that were actually correct, while recall finds the proportion of actual positives that were identified correctly. Their formulas are below, where TP represents true positives, TN represents true negatives, FP represents false positives, and FN represents false negatives:

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

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

Accuracy can be described as the number of predictions the model got correct. The formula is below:

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

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

4.2 ResNet34

The following table summarizes F1 score, accuracy, precision, and recall for the training set, validation set, and testing test:

Set	F1 score	Accuracy	Precision	Recall
Training	0.7306	0.7306	0.7307	0.7306
Validation	0.6271	0.6264	0.6268	0.6274
Testing	0.7185	0.7179	0.7182	0.7189

Table 1: Metrics for ResNet34. Metrics for training and validation taken at the 19th epoch due to early stopping.

The following graphs show the model's training and validation accuracy and training and validation loss over 19 epochs:

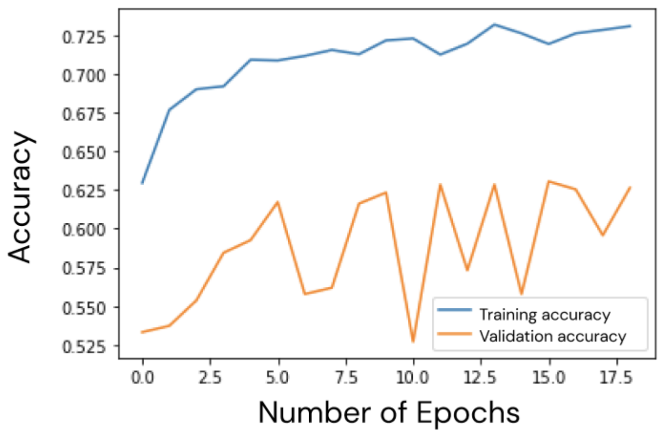

Figure 3: The training accuracy and validation accuracy of the model using ResNet34.

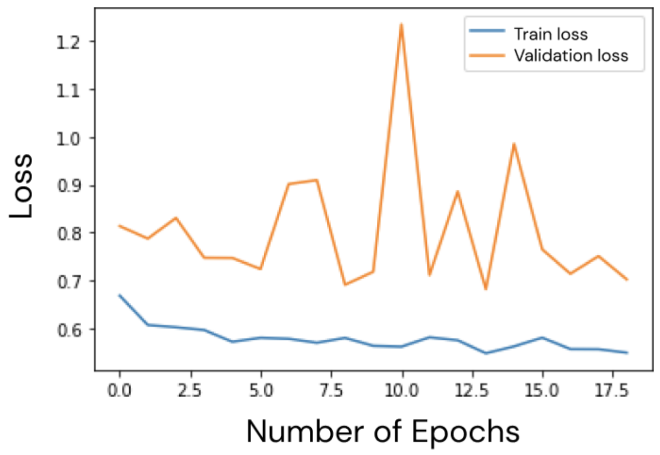

Figure 4: The training loss and validation loss of the model using ResNet34.

As seen in Table 1, the F1 score and accuracy of the model made with ResNet34 is fair. However, it does not perform well enough to be deployed for real world use. One point of concern is the lack of stability in validation accuracy and loss. This could be due to the fact that the model did not train for enough epochs. Although early stopping was implemented, the patience could be increased to a greater number, say 20, instead of 5.

4.3 ResNet50

The following table summarizes F1 score, accuracy, precision, and recall for the training set, validation set, and testing test:

Set	F1 score	Accuracy	Precision	Recall
Training	0.7527	0.7533	0.7528	0.7526
Validation	0.7478	0.7472	0.7474	0.7482
Testing	0.6764	0.6761	0.6746	0.6782

Table 2: Metrics for ResNet50. Metrics for training and validation taken at the 15th epoch due to early stopping.

The following graphs show the model's training and validation accuracy and training and validation loss over 15 epochs:

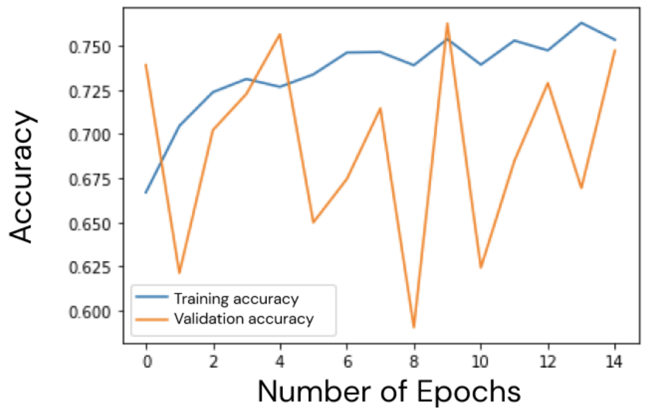

Figure 5: The training accuracy and validation accuracy of the model using ResNet50.

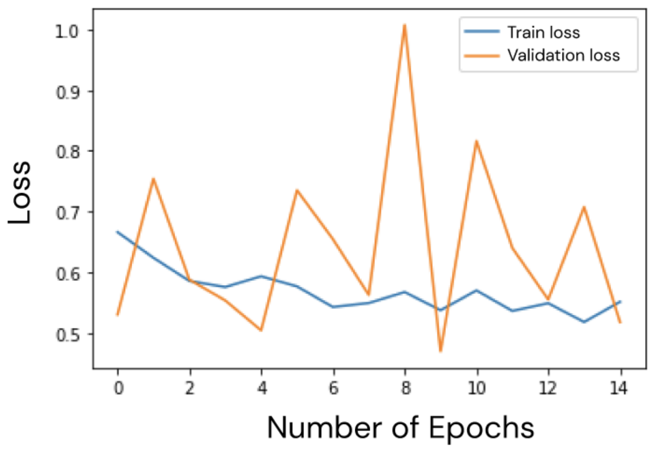

Figure 6: The training loss and validation loss of the model using ResNet50.

The same points made in the previous section can be applied to the model using ResNet50. The F1 score and accuracy of the model are fair but do not indicate that it can be used in the real world. Once again, the lack of stability in validation accuracy and loss is a point of concern. The same solution, increasing the value for patience, could be used to address this problem.

4.4 Comparison

Both of these models lack stability in validation accuracy and loss and do not perform well enough for real world use. An unexpected result was that model using ResNet34 performed better during testing than the model using ResNet50. However, the model using ResNet50 perfomed better during training and validation. These mixed results lead to an inconclusive result. Because one model did not consistently perform better than the other, it cannot be said with confidence that one pre-trained model is better than the other.

The reason that the results were mixed may have been that the model was not trained for long enough. Again, patience should be increased in order to ensure that validation loss is stable.

5 Conclusion

In this paper, the effect of the ResNet's size on the performance of the model was explored. Two models using different sizes of ResNet (34 and 50) were trained and tested on a dataset of images with invasive ductal carcinoma. This paper attempted to address the current gap in knowledge regarding the role of ResNet size in a model's performance in cancer identification. The first step in addressing this knowledge gap was obtaining a dataset publicly available on Kaggle that included images that either contained IDC or did not. Then, the data was downsampled to address class imbalance. Next, the necessary APIs and libraries were imported, as well as necessary parameters being specified to set the model up. The data was then pre-processed and augmented to prevent overfitting, and the model was finally trained and evaluated.

The model using ResNet34 performed better in testing, while the model using ResNet50 peformed better during training and validation. Additionally, the validation accuracy and validation loss in both models was very unstable. Due to these reasons, the results of this project are inconclusive, and further research and fine-tuning of these models would be beneficial. When the results of this research become conclusive, it can be applied to healthcare, especially the diagnosis of diseases. Studying the performances of different sized pre-trained models is important because data is difficult to access in healthcare, and pre-trained models reduce the amount of data needed to train models.

5.1 Future Work

One way to address the current limitations in this project would be to increase the patience and let the model run for many more epochs than it did in this project. There are many possibilities for future work with this project. One procedure that could be implemented during data pre-processing is k-fold cross validation. This procedure could be implemented to ensure that there are no biases in the dataset. Expanding to working with more sizes of ResNets could also be explored, as there are ResNets of sizes 18, 34, 50, 101, and 152.

References

[1] H. Sung, J. Ferlay, R. L. Siegel, M. Laversanne, I. Soerjomataram, A. Jemal, and F. Bray, "Global cancer statistics 2020: Globocan estimates of incidence and mortality worldwide for 36 cancers in 185 countries," *CA: a cancer journal for clinicians*, vol. 71, no. 3, pp. 209–249, 2021.

[2] A. Cruz-Roa, A. Basavanhally, F. González, H. Gilmore, M. Feldman, S. Ganesan, N. Shih, J. Tomaszewski, and A. Madabhushi, "Automatic detection of invasive ductal carcinoma in whole slide images with convolutional neural networks," in *Medical Imaging 2014: Digital Pathology*, vol. 9041, p. 904103, International Society for Optics and Photonics, 2014.

[3] D. L. Page, "Special types of invasive breast cancer, with clinical implications," *The American journal of surgical pathology*, vol. 27, no. 6, pp. 832–835, 2003.

[4] S. Albawi, T. A. Mohammed, and S. Al-Zawi, "Understanding of a convolutional neural network," in *2017 International Conference on Engineering and Technology (ICET)*, pp. 1–6, Ieee, 2017.

[5] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.

[6] S. A. Alanazi, M. Kamruzzaman, M. N. Islam Sarker, M. Alruwaili, Y. Alhwaiti, N. Alshammari, and M. H. Siddiqi, "Boosting breast cancer detection using convolutional neural network," *Journal of Healthcare Engineering*, vol. 2021, 2021.