# Detecting Breast Cancer From Images with a CNN

Amy McCaughan, Breanna Moore, Kenny Vangemert

## Overview of Project

For this project we are trying to classify an image of a breast as cancerous or not by using a CNN. This can help doctors make better and more informed decision when diagnosing a patient. It can also provide a second opinion to help when doctors are unsure of the diagnosis.

We want the model to be accurate because this decision by the model can drastically impact a patient if it is wrong. If given a false positive patients can undergo expensive treatment they do not need, and if a false negative then people could die from this fatal mistake by the model.

The dataset consists of labeled train images and unlabeled test images. Train and test images are composed of multiple folders with patients' numbers. Each patient should have 4 images in it for upper, lower, left, and right image of the breast. However, some images do not have all 4 images.

## Challenges and Work Done

- There is a disproportionate amount of noncancerous data compared to cancer data. 1158 cancerous images, and 53555 noncancerous images.
- Not all DCM files are in the proper greyscale format.
- DCM files storing the mammograms are extremely large and required transformations.
- Finding an optimal number of layers to use in the model, too many and the model would take too long to run, but too little meant a poor model.
- Finding an optimal learning rate for the model since too low or too high of a learning rate caused the model to have very poor metrics.
- Model training takes a long time to perform, tweaking and tuning the model took an extensive amount of time because of so.
- Keras utility class is used to increase speed of grabbing the data and feeding it to the model, as well as perform the tasks mentioned below.
- Normalization and greyscale conversion was done to allow for coherency in pixel values, as well as increase the speed the model trains at.
- Instead of using all 54713 images, we split it to use all 1158 cancer positive images and grab a random 1158 non cancer positive images to allow an equal distribution for the model to learn on.

## Methods/Tools

- TN Tech Open OnDemand was used to store the data since the data size was over 300GB, it also allowed the use of Jupyter Notebooks for coding and running the model.
- A CNN was utilized for this since it is one of the most effective models for image data.
- TensorFlow is used to utilize GPU power and to access the CNN components needed.
- PyDicom is used to access the information needed regarding our DCM files. Allows access to pixel values and color modes.

## Results

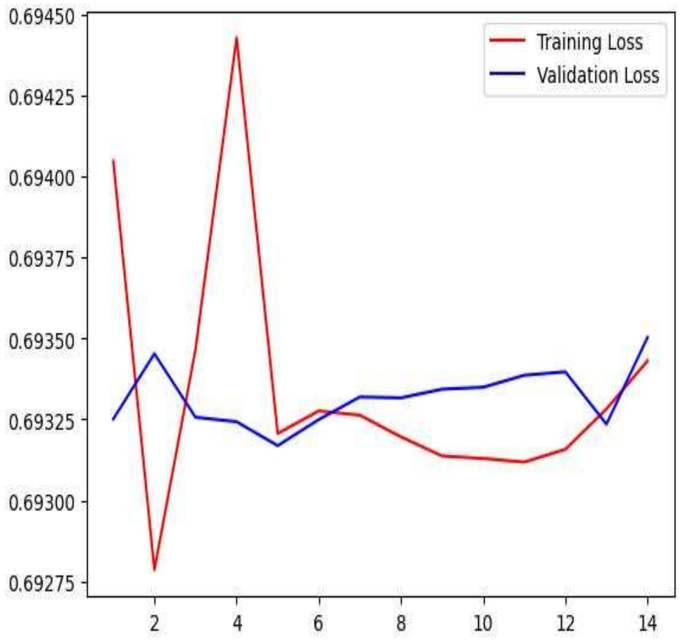
- When looking at the metrics, it is important to note the model was given 3 different thresholds to calculate at, however, due to some weird error, only 0.5 produced any real result, so for the calculations, they will all be based off a threshold of 0.5.
- Precision, what percentage of positive breast cancer predictions are correct: 0.3062
- Recall, what percentage of actual positive breast cancer cases were correctly predicted: 0.3621
- F1, a combination of precision and recall and shows the model's performance: 0.3314
- Accuracy, overall correctness of the model: 0.5018
- When used on the test data, the model had predicted 0.49373 for the left breast, and 0.49372 for the right breast. These scores are very good and way higher than what was being anticipated.

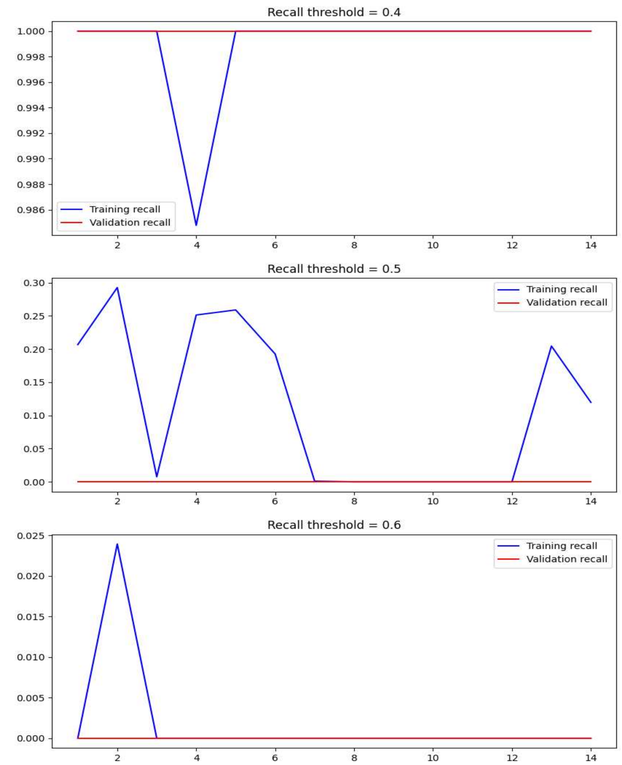
| Layer (type) | Output Shape | Param # |
|---|---|---|
| conv2d_6 (Conv2D) | (None, 252, 252, 32) | 832 |
| conv2d_7 (Conv2D) | (None, 248, 248, 64) | 51264 |
| max_pooling2d_3 (MaxPooling2D) | (None, 124, 124, 64) | 0 |
| dropout_5 (Dropout) | (None, 124, 124, 64) | 0 |
| conv2d_8 (Conv2D) | (None, 120, 120, 64) | 102464 |
| conv2d_9 (Conv2D) | (None, 116, 116, 128) | 204928 |
| max_pooling2d_4 (MaxPooling2D) | (None, 58, 58, 128) | 0 |
| dropout_6 (Dropout) | (None, 58, 58, 128) | 0 |
| conv2d_10 (Conv2D) | (None, 54, 54, 128) | 409728 |
| conv2d_11 (Conv2D) | (None, 50, 50, 256) | 819456 |

...
Total params: 1596929 (6.09 MB)
Trainable params: 1596929 (6.09 MB)
Non-trainable params: 0 (0.00 Byte)

## Discussion

- Since the main challenge of the project is to create a model that can identify breast cancer, it is important to dive a bit deeper into the prediction results.
- As stated, the model predicted 0.49373 for the left breasts chance at having cancer, and 0.49372 for the right breast. If this were a real-life situation where this model was being used, this would most likely be a little alarming for the doctors, since there is a 0.49% chance whether the patient has cancer or not. Although this number is not extremely high like 0.80+, one may not think of too much importance to do more testing and research on the patient. But this is where an issue may come into play, since models are not going to be 100% accurate all the time, when a relatively higher number, like in this instance of 0.49% is shown for a patient's chance of cancer, it would not be a bad idea to investigate it more. If the number was significantly lower like 0.2 or even less, then it may not be a big deal and can almost be guaranteed that the patient does not have cancer. However, when working with something as dangerous and life changing as cancer, any sort of sign or prediction of cancer being present should always be taken seriously.




## Conclusion/Recommendations

- To conclude, the model turned out to be rather decent providing a 0.49% chance of the test patient having breast cancer. The model's metrics were also not horrible seeing an F1 score of 0.3314 and a binary accuracy score of 0.5018.
- For future recommendations, it would be way easier if a more powerful computer was used to speed up the training time. Many hours were spent just trying to train the model to find the best parameters, yet this was incredibly difficult due to the model taking about 2-3 hours just to get to a point where it could be seen if things needed to be tweaked or not. If a better computer was provided, a higher number of epochs could have been used in training allowing more errors and faults to be caught, it also could allow a more in-depth model of more layers which could potentially fix some issues that arose.

## References/Acknowledgements

- We would like to thank Dr. Eberle for his guidance and support throughout the semester, as well as Dr. Renfro for letting us utilize TN Tech Open OnDemand for the project.

- There are no references used in this research poster.