Machine Learning Engineer Nanodegree

Capstone Project

Tanush Goel

July 5th, 2019

I. Definition

Project Overview

Invasive ductal carcinoma (IDC) is the most common form of invasive breast cancer, a cancer that begins growing in a milk duct and soon invades the tissue of the breast outside the duct.

The project was to create a convolutional neural network to classify an image of a breast tissue sample as being positive or negative for invasive ductal carcinoma.

In order to detect cancer, a tissue sample is put on a glass slide, and a pathologist has to then examine this slide under a microscope. The pathologist needs to look through very large regions where there may be no cancer in order to ultimately find malignant areas. These glass slides can now be digitized, allowing computer vision to help speed up a pathologist's workflow, provide diagnosis support, and increase patient survival rate.

Problem Statement

More women are diagnosed with breast cancer than any other cancer, besides skin cancer. As with any breast cancer, there may be no signs or symptoms. Invasive ductal carcinoma represents 80 percent of breast cancer cases/diagnoses. A mammogram may reveal a suspicious mass, which will lead to further testing including biopsies, which involve taking out some or all of the abnormal-looking tissue for examination by a pathologist (a doctor trained to diagnose cancer from biopsy samples) under a microscope. More than 1 million women have breast biopsies each year in the United States. About 20 percent of these biopsies yield a diagnosis of breast cancer. This year, an estimated 268,600 women in the United States will be diagnosed with invasive breast cancer and 2,670 men in the United States will be diagnosed with breast cancer. It is estimated that 42,260 deaths (41,760 women and 500 men) from breast cancer will occur this

year. A convolutional neural network trained on infected and normal tissue samples could be able to help doctors classify and understand the images better. It could also tell the doctor which cases are more important to observe allowing the patient to be treated quicker, increasing the patient's chances of survival.

Metrics

The model can be measured by simply the accuracy (total correctly predicted over total predicted) or f1-scores of both classes (average of precision and recall). It could also be measured by the sensitivity (true positives/(true positives + false negatives)) and specificity (true negatives/(true negatives + false positives)) of both classes as it is a medical binary classification problem, but the positive class would matter more because letting a malignant case go without treatment is much worse than having a non-malignant case go for treatment. However, if a non-malignant case is predicted to be positive for IDC, the patient may have to go for unnecessary and expensive surgery to remove malignant sites.

II. Analysis

Data Exploration

1.64GB breast histopathology image dataset by Kaggle:
https://www.kaggle.com/paultimothymooney/breast-histopathology-images

Download Link:
https://www.kaggle.com/paultimothymooney/breast-histopathology-images/downloads/breast-histopathology-images.zip/1

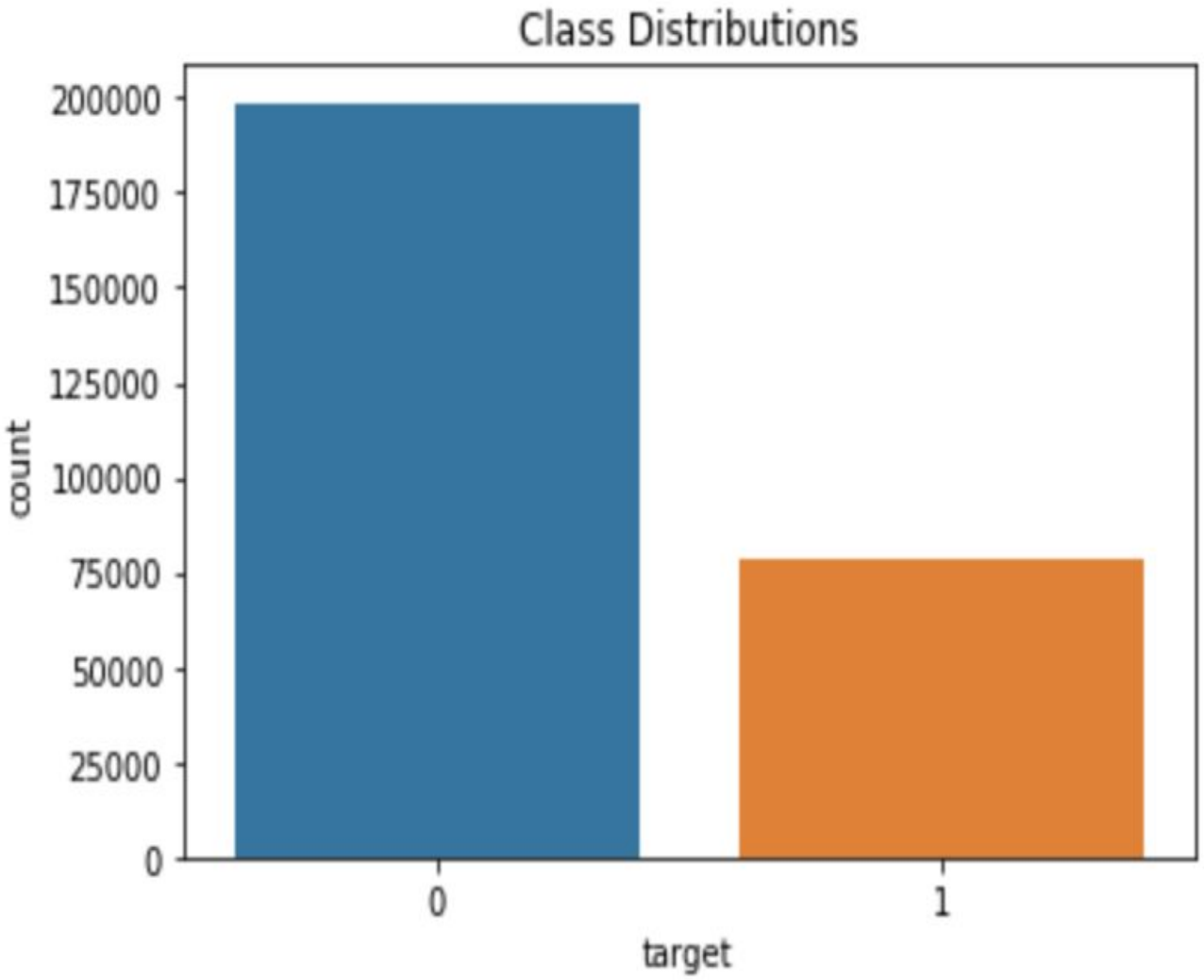
The original dataset consisted of 162 whole mount slide images of Breast Cancer (BCa) specimens scanned at 40x. From that, 277,524 patches of size 50 x 50 were extracted (198,738 IDC negative and 78,786 IDC positive).

The original files are located here:
http://gleason.case.edu/webdata/jpi-dl-tutorial/IDC_regular_ps50_idx5.zip

Citation:
https://www.ncbi.nlm.nih.gov/pubmed/27563488
http://spie.org/Publications/Proceedings/Paper/10.1117/12.2043872

Exploratory Visualization

The class distributions are visualized in the figure below.



277524 total images - 198,738 IDC negative images and 78,786 IDC positive images which is about a 2.5 : 1 distribution ratio and is a huge class imbalance

Algorithms and Techniques

The algorithm is a convolutional neural network, which is a state-of-the-art algorithm for most image processing tasks including classification. It needs a large amount of training data compared to other approaches, and this Kaggle dataset is large enough. Convolutional neural

networks are regularized versions of multilayered perceptrons that can take advantage of the spatial arrangement of the data, like the pixel values in an image. They extract features via the learnable filters/kernels among the model's convolutions (the core building block of a CNN). The feature map matrix is then converted into a vector. Fully connected layers are combined with these features so the model is able to understand them. A final 2 unit output fully connected layer with a softmax activation function allows the model to classify an image as being either IDC positive or IDC negative.

The model can output assigned probabilities for either class to reduce the number of false positives using a threshold, but this would also increase the number of false negatives.

The following can be tuned to optimize the classifier:

- Classification Threshold / Class Weights
- Preprocessing Parameters
    - Preprocessing Function (Inception, ResNet, VGG, custom, none)
    - Data Augmentation Parameters
- Neural Network Architecture
    - Number of Layers
    - Layer Types (Convolution, Pooling, Dropout, Dense/Fully Connected)
    - Number of Filters/Units
- Training Parameters
    - Epochs
    - Batch Size
    - Optimizer
    - Learning Rate

During training, the datagen class pulls data from defined folders in small processed batches to load into RAM and feeds it into the model for training, validation, or testing. The training is done with mini-batch gradient descent.

Benchmark

Most other models trained on this data have about 85% testing accuracy on about 15,000 images. The models are simple keras CNN's of 2-4 convolutional layers, some with some forms of data augmentation.

https://www.kaggle.com/ambarish/breast-histopathology-analysis - This model is a 3-layer keras sequential CNN that achieves a 85.52% validation accuracy on 18,000 images.

Other models are evaluated in these links:
https://www.kaggle.com/dineepthomas26/predict-idc-in-breast-cancer-histology-images
https://www.kaggle.com/vbookshelf/part-1-breast-cancer-analyzer-web-app
https://www.kaggle.com/ambarish/breast-histopathology-pytorch
https://www.kaggle.com/sayantandas30011998/invasive-ductal-carcinoma-classifier-using-fastai

III. Methodology

Data Preprocessing

To prepare the data, the images were transferred into train, validation, and test folders. Augmented duplicates of the IDC positive class were made as it is underrepresented. The InceptionV3 preprocessing function in keras was then used on all the images as they were fed into the model. This function divides the pixel values by 255, subtracts 0.5, and multiplies by 2. It is totally different from the preprocessing of other models such as Resnet and VGG. There is no mean subtraction nor RGB to BGR. All pixel values with this function should be between -1 and 1. An image size of 100 was chosen, which was double the size of the taken image and large enough to fit into a transfer learning model. This image size also led to the best resulting accuracy of the benchmark model.

Implementation

Steps:

1. Imports and Dependencies - all necessary packages were installed
    a. Tensorflow 2.0 has keras built in, but version 1.14.0 was used
2. Data Extraction - data was uploaded and unzipped
    a. Data can be uploaded from Kaggle directly using a Kaggle API key
3. Copy Data Into Directories - images were moved into two subfolders (1 for the positive class and 0 for the negative class) of a directory called ‘all_images_dir’
    a. Images are extracted from their patient ID folder > class folder and moved into all_images_dir folder > class folder
4. Make Dataframe of Image Information - dataframe made of image ID’s, patient ID’s, and target (IDC positive or IDC negative as 1 or 0 respectively)
5. Train/Test/Valid Split - dataframe split into train, valid, and test images
6. Move Train/Test/Valid Images Into Respective Directories - images from dataframe moved into train, valid, and test folders

    a. Images are extracted from their patient ID folder > class folder and moved into train > class folder, valid > class folder, test > class folder
7. Make Augmented Images of Underrepresented Class (IDC Positive Class) - augmented duplicate images of the IDC positive class in the train folder are made to help overcome bias from class imbalance
8. Create Train/Valid/Test Batches - train, valid, and test batches are made with the keras datagen class
    a. Inception preprocessing function is used
    b. Batch size is set to 500
    c. Test Batches should NOT be shuffled
9. Train Benchmark Model - benchmark CNN built and trained
    a. Class weight can be increased for the positive class to make the model more sensitive towards it
10. Evaluate Benchmark Model - benchmark model evaluated on test data
    a. Accuracy, balanced accuracy, F1 score, precision, and recall calculated
    b. Confusion matrix shown

The transfer learning model did not perform better than a benchmark CNN, but if one is wanted, most keras transfer learning models require an image size of 75*75 or larger, so a larger image size must be fed into the benchmark model first and the weights of the model added into a transfer learning model.

When using the Pillow library, make sure to set "LOAD_TRUNCATED_IMAGES" to "True" or else many errors will occur when images are opened: This can be done with this line of code:
```
ImageFile.LOAD_TRUNCATED_IMAGES = True
```

Run Tensorflow as the backend for Keras. If what is defined in the config file is not Tensorflow, an override is possible by defining the environment variable "KERAS_BACKEND" with this line:
```
KERAS_BACKEND = tensorflow python -c "from keras import backend"
```

When importing packages from Keras, upload it from keras directly and not from tensorflow.python.keras unless Tensorflow 2.0 is being used.

Refinement

Augmented images were made of the IDC positive class because of the large class imbalance among the data. Data augmentation parameters were tweaked a lot, but only a minor improvement in accuracy was made. Same padding in the CNN convolutional layers greatly improved accuracy. Image size had a seemingly large impact on accuracy, and a size of 100 by 100 worked the best for me. A larger dropout prevented quick overfitting of the training data.

More layers of 2D convolutional layers worked better than simply more filters. The batch size was changed a lot, and a large size of 500 took good steps and convergence resulted quickly. Other transfer learning models incorporating the benchmark model's weights were tried, but this resulted in a significantly worse accuracy.

IV. Results

Model Evaluation and Validation

During development, a stratified validation set and testing set was used to evaluate the model. The final architecture and hyperparameters were chosen because they performed the best among the tried combinations.

Model Architecture:

- 9 convolutional layers - Pooling & Dropout layer after every 3 convolutional layers
    - First 3 convolutional layers have 16 filters each
    - Second 3 convolutional layers have 32 filters each
    - Third 3 convolutional layers have 64 filters each
- Filter sizes = 3*3
- Padding = 'same'
- Stride = 1
- Flatten layer turns output of convolutional layers into a vector
- First dense layer has 256 units
- Output layer has 2 units corresponding to the IDC positive & IDC negative classes
- Batch size = 500
- Epochs = 30
- Each epoch takes about 3-6 minutes on a Tesla K80 GPU provided by Google

To verify the robustness of the model, the model was tested on augmented test images (rotated, flipped vertically and/or horizontally, shifted vertically and/or horizontally, zoomed in on), and the accuracy was close to the same. Training on augmented images might have helped with this more, allowing the model to be better trusted.

Justification

The CNN had an accuracy of 90.3% on a 10,000 image stratified testing set and a highest accuracy of 91.2% on a 1000 image stratified validation set. Compared to the benchmark final validation accuracy was 85.52% on an 18,000 image validation set, this is a decent improvement.

This accuracy is still not significant enough to have solved the problem however. For the hundreds of thousands of people who are tested for IDC every year, this is not satisfactory in my opinion. However, there are multiple slides collected from each person, so if the model misses a malignancy in one slide may catch it in another. This unfortunately is a chance patients should not risk their lives on. The model should be used to help doctors by bringing up more important cases allowing for a faster workflow and the patient to be treated quicker, and maybe provide some diagnosis help.

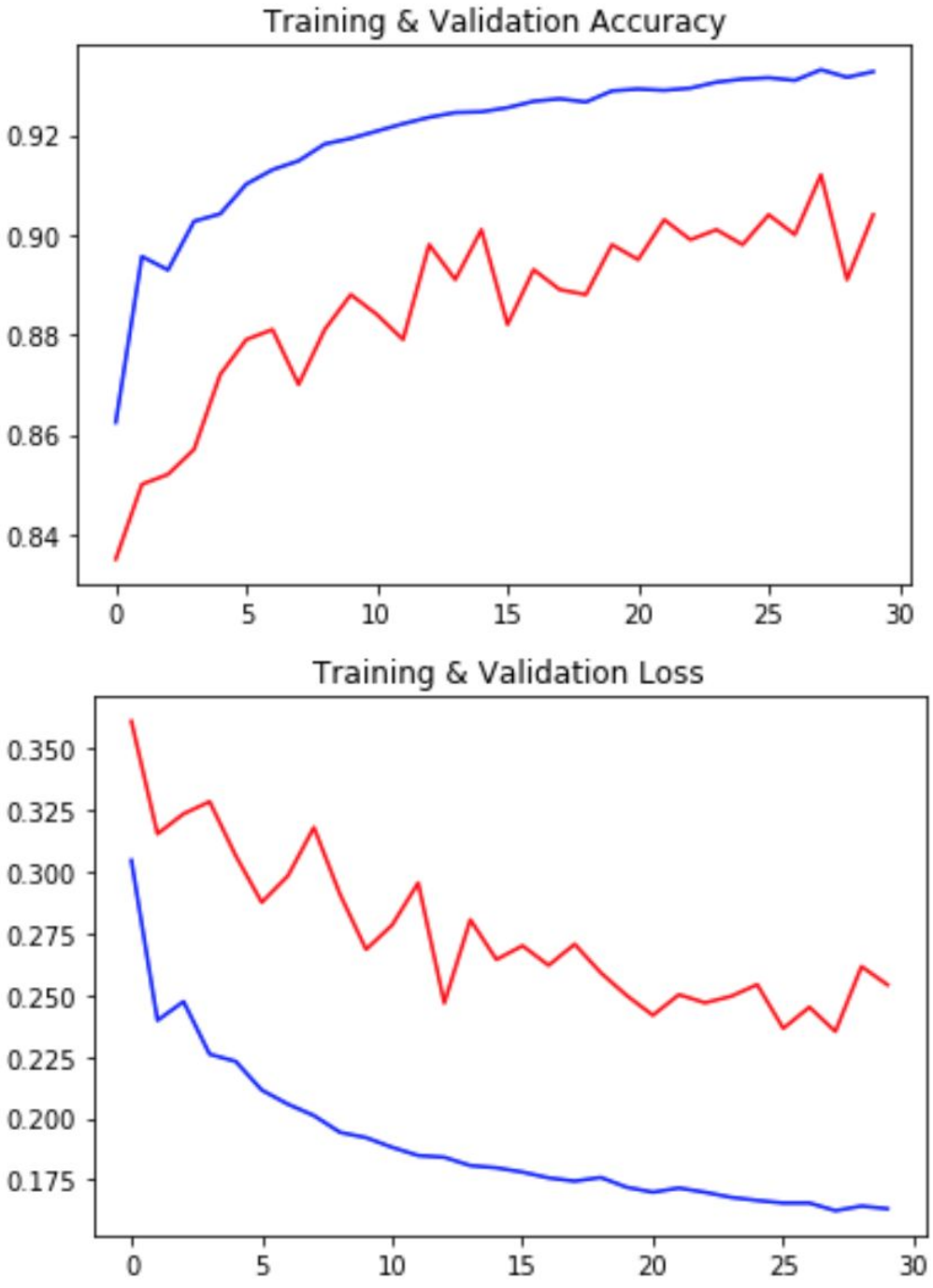
V. Conclusion

Free-Form Visualization

Model Summary:

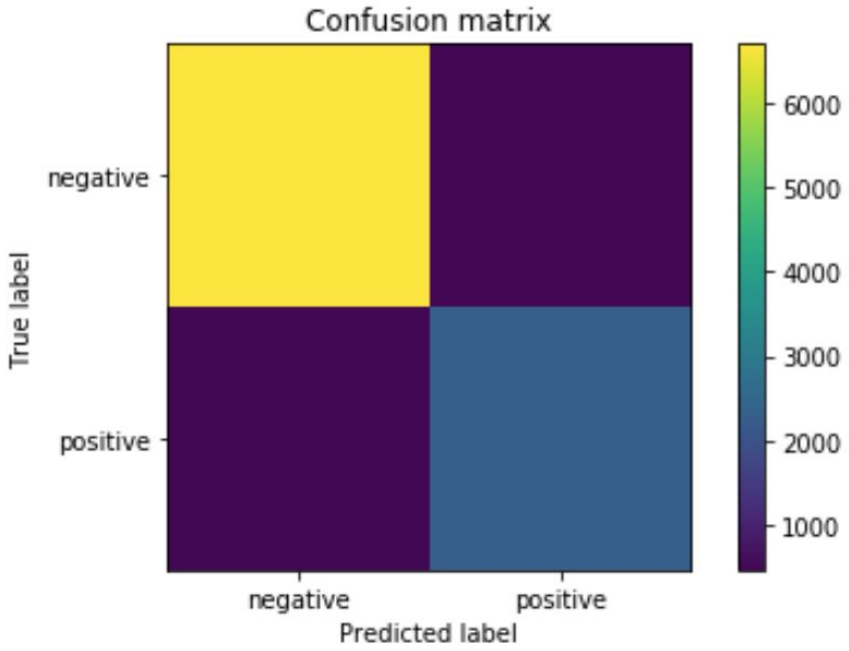
Layer (type)	Output Shape	Param #
conv2d_10 (Conv2D)	(None, 100, 100, 16)	448
conv2d_11 (Conv2D)	(None, 100, 100, 16)	2320
conv2d_12 (Conv2D)	(None, 100, 100, 16)	2320
max_pooling2d_4 (MaxPooling2	(None, 50, 50, 16)	0
conv2d_13 (Conv2D)	(None, 50, 50, 32)	4640
conv2d_14 (Conv2D)	(None, 50, 50, 32)	9248
conv2d_15 (Conv2D)	(None, 50, 50, 32)	9248
max_pooling2d_5 (MaxPooling2	(None, 25, 25, 32)	0
conv2d_16 (Conv2D)	(None, 25, 25, 64)	18496
conv2d_17 (Conv2D)	(None, 25, 25, 64)	36928
conv2d_18 (Conv2D)	(None, 25, 25, 64)	36928
max_pooling2d_6 (MaxPooling2	(None, 12, 12, 64)	0
dropout_3 (Dropout)	(None, 12, 12, 64)	0
flatten_2 (Flatten)	(None, 9216)	0
dense_3 (Dense)	(None, 256)	2359552
dropout_4 (Dropout)	(None, 256)	0
dense_4 (Dense)	(None, 2)	514

Training and Validation Plots:



Rough improving trend in validation accuracy is visualized until around 30 epochs. Over 30 epochs in training leads to gradual overfitting.

Confusion Matrix:

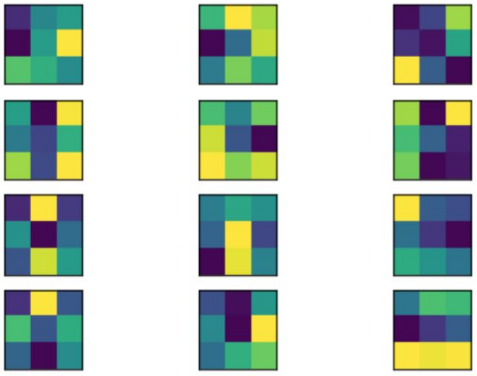



```
[6696,  465]
[ 504, 2335]
```

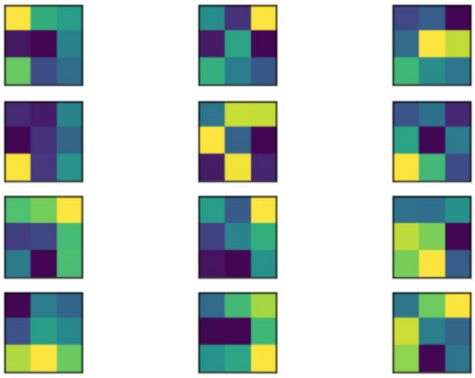
Classification Report:

	precision	recall	f1-score	support
negative	0.93	0.94	0.93	7161
positive	0.83	0.82	0.83	2839
accuracy			0.90	10000
macro avg	0.88	0.88	0.88	10000
weighted avg	0.90	0.90	0.90	10000

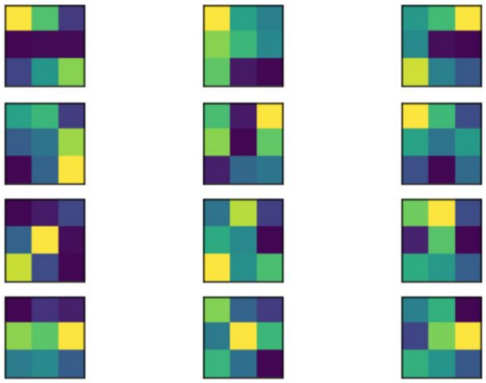
Filters of 3rd convolutional layer (4 for each in RGB):



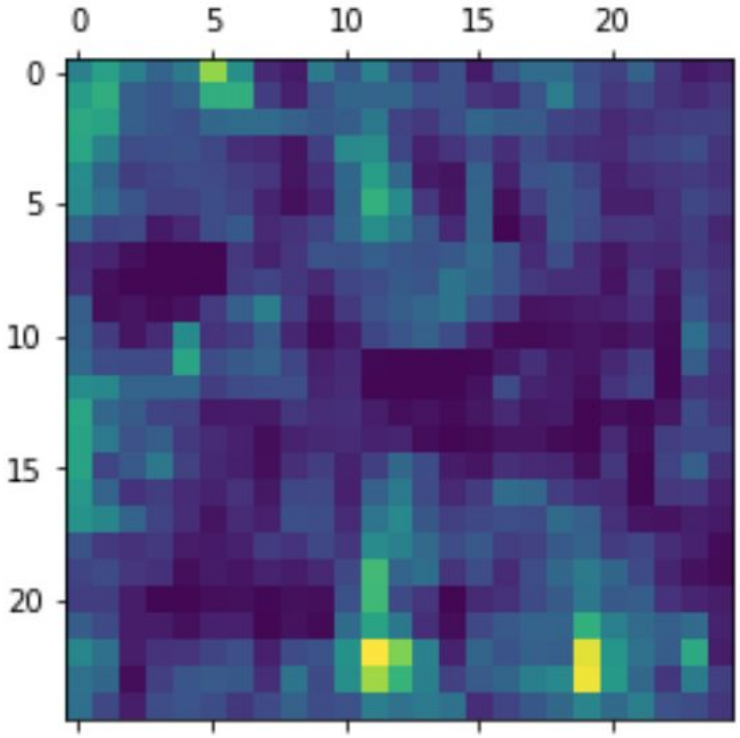
Filters of 6th convolutional layer (4 for each in RGB):



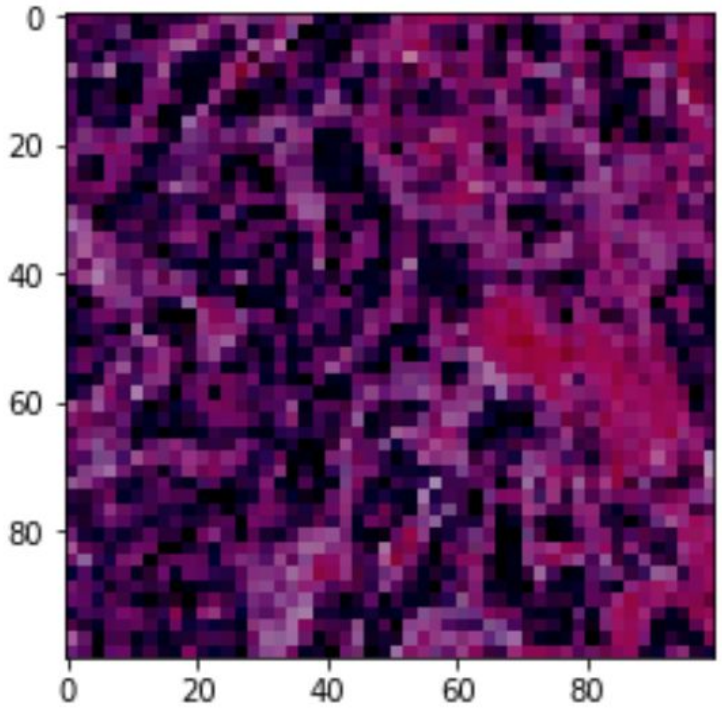
Filters of 9th convolutional layer (4 for each in RGB):



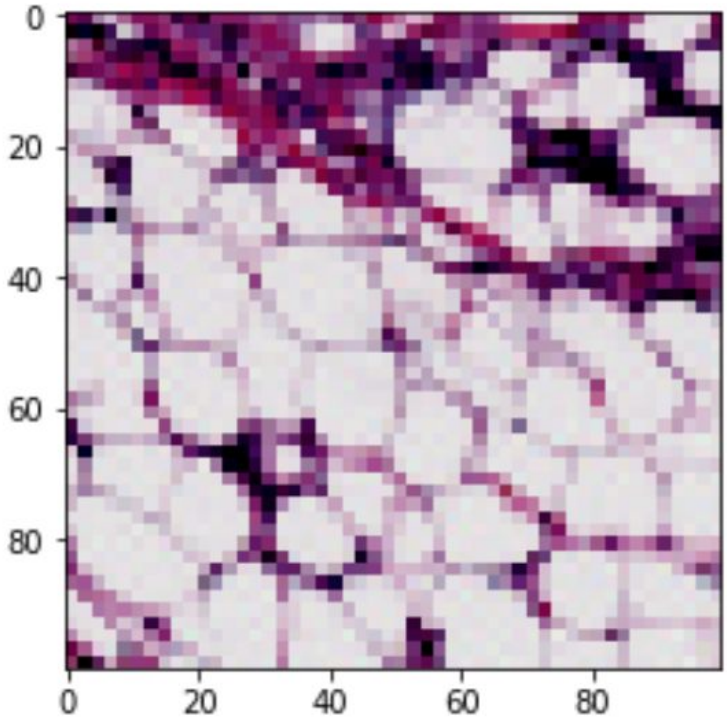
Class Activation Maps (CAM):



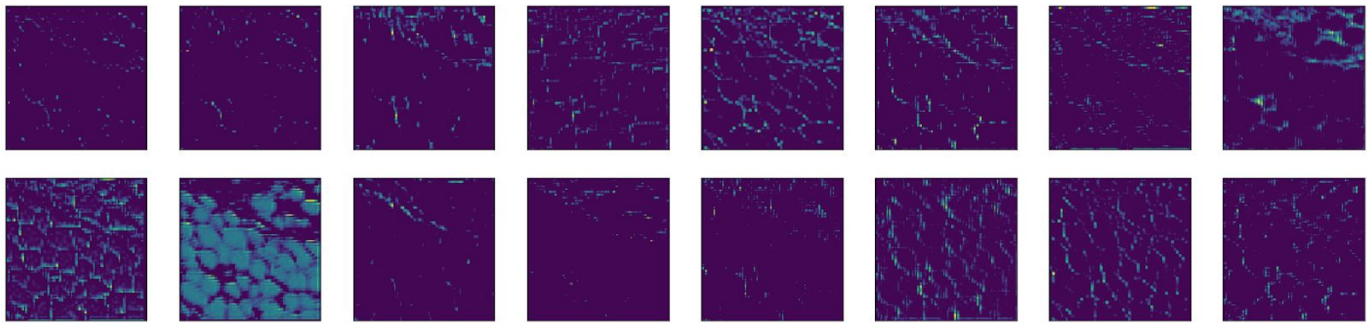
W0726 20:15:02.121720 139884069193600



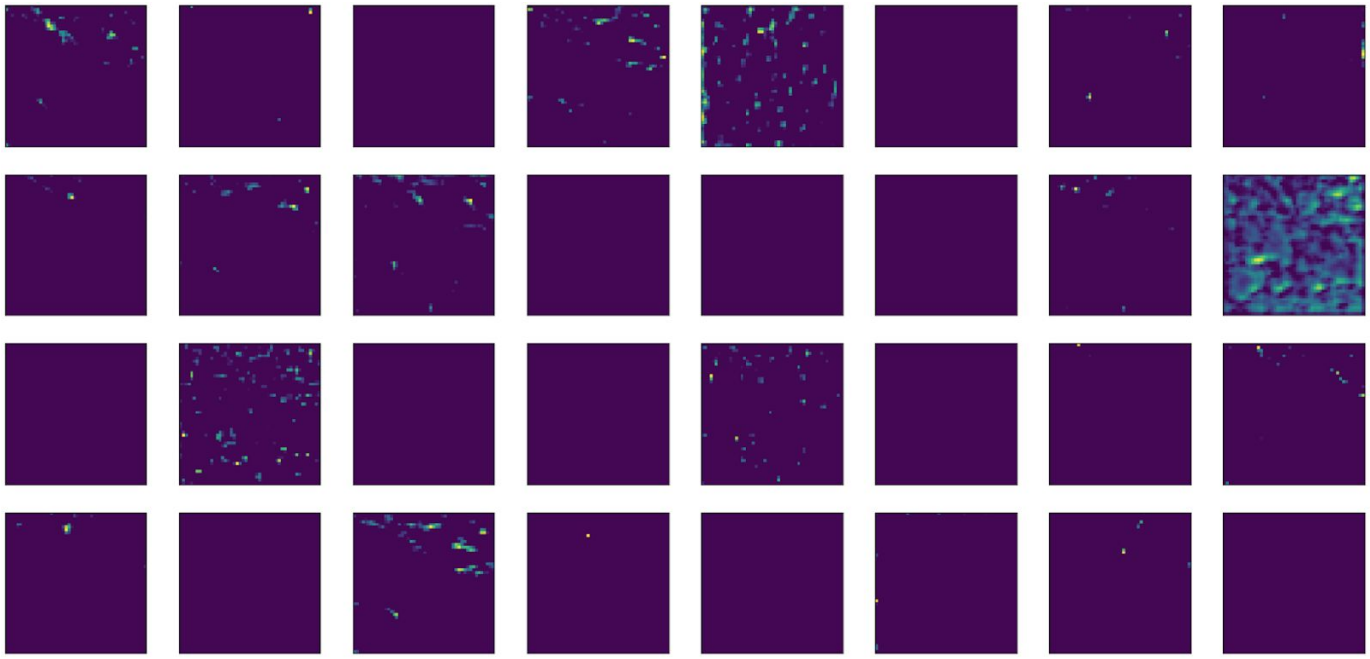
```
IDC positive
Predicted Negative Probability: 37.46%
Predicted Positive Probability: 62.54%
```



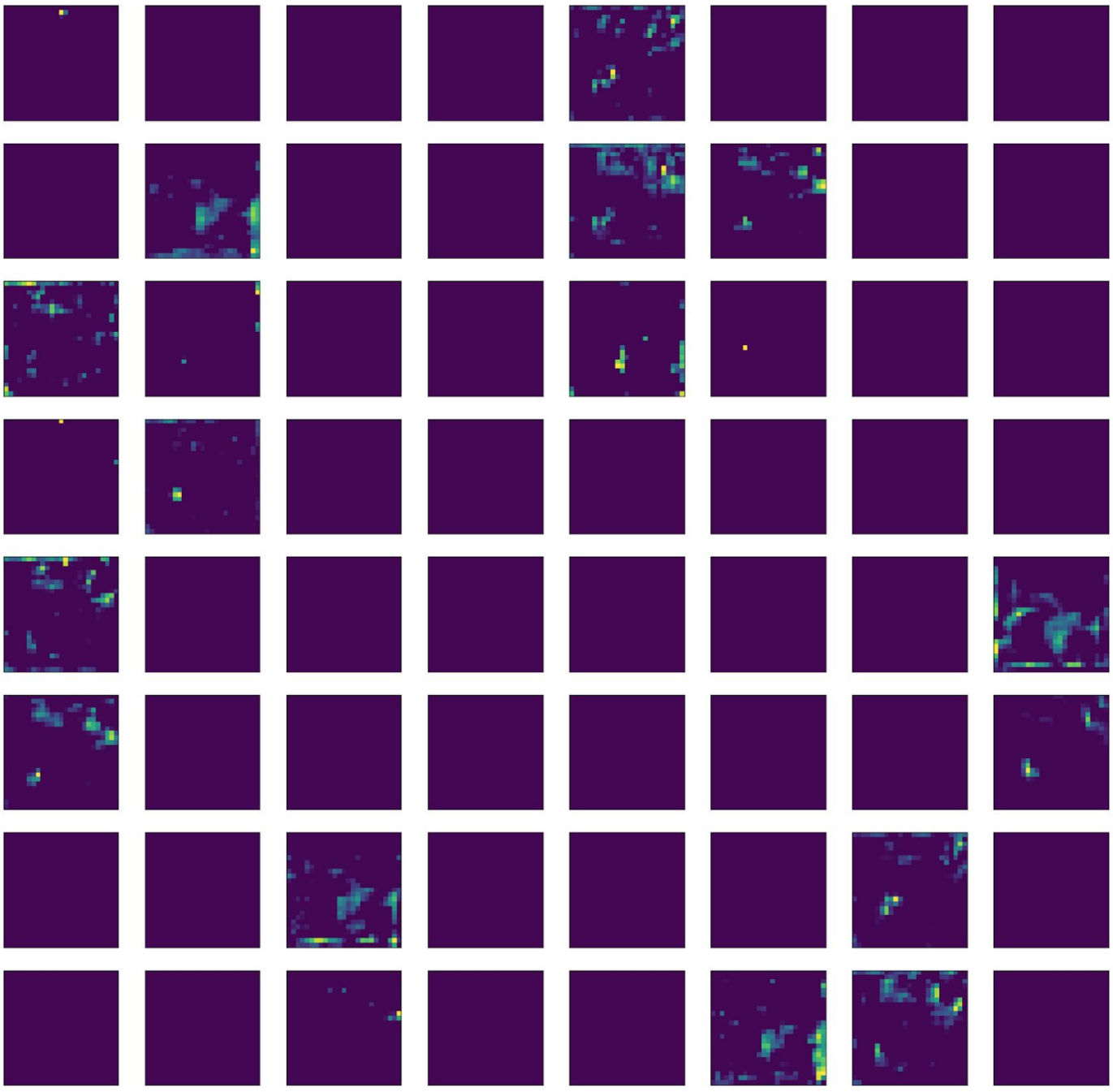
High Level:



Intermediary Level:



Low Level:



Reflection

1. Download and Preprocess the Data
2. Train a CNN Model
3. Evaluate the Results
4. Use for Predictions

The most difficult aspect to the project was how to preprocess/augment training samples. There are so many ways to do this making it impossible to test them all. It is very interesting to see the class activation maps of the model as it extracts lower level features from the image. The whole process behind how the model is able to classify the image is revealed through these activations which is incredible to observe.

Improvement

Other transfer learning models better suited to the task could be used to increase accuracy and speed up convergence. More data could also be used, including histopathologic cancer outside of the breast area. This Kaggle dataset could also be incorporated from the Histopathologic Cancer Detection competition: https://www.kaggle.com/c/histopathologic-cancer-detection

On top of this, other hyperparameters could be tweaked and model architecture improved. Data augmentation parameters and methods could especially be improved.