Cancer Detection with CNNs

Miguel Antonio Oliveira Figueiredo 20231436
Miguel Maria Cappelle Teixeira Fernandes Pega 20232022
Pedro Miguel Albuquerque Gomes 20231126
Simona Costache 20231983

Abstract: The following essay explores the capabilities of Convolutional Neural Network (CNN) models and their ability to help identify cancer of a given set of images. There are two steps for this project, consisting of binary classification and multiclass classification. The methodology consisted of collecting and exploring data, preprocessing data, defining the model, shaping its architecture and training it. The models were then evaluated and optimizations were applied to increase performance and obtain better results against test data. The models which obtained the best result were found to be with data undersampling and using optimization techniques like class weights, regularization, batch normalization, layer dropouts and data augmentation. Future work still needs to be done in order to try different network topologies by focusing mainly on the stage 1 of the project which shows room for further improvement.

I. Introduction

Breast cancer is the second most diagnosed cancer in women, affecting 1 in every 8 women in the US, according to the National Cancer Institute, representing 16% of all new cancer cases and 7% of all the cancer deaths in 2024. Early detection is crucial, as women whose breast cancer is identified at an early stage have a 93% or higher survival rate in the first five years. These early detections are often achieved through screening mammograms, which show the importance of having tools to help read, interpret and detect these anomalies.

The task at hand can be formulated as a binary classification problem, where a set of 7909 images distributed between malignant and benign cancer types will be used as input. This project aims to develop a CNN model to classify histological images, firstly into benign or malignant images, and, in a second stage, extend this model to a multi class classification, based on breast cancer type.

II. Methodology

A. Data Collection and Exploration

BreaKHis dataset was exploited in this project, which is a comprehensive collection of breast cancer histological images. It is composed of 7,909 images, divided between benign and malignant classes, with each of them then divided into 4 types of breast cancer. For each type of cancer, a set of subjects were analysed and submitted to radiology tests resulting in a set of images which were decomposed by their magnification, from 40X to 400X. From a general standpoint, their Cancer Class was not evenly split, in favour of malignant, with 68.6% against 31.4% benign subjects. As seen below Ductal Carcinoma is the top contributor for image population accounting 43.6% of all the images (3451 out of 7909) followed by Fibroadenoma and Mucinous Carcinoma with 12.8% and 10.0% respectively. Finally, 82 subjects were identified in this dataset with an evenly distributed image magnification.

This dataset also includes metadata such as image paths, classification labels, cancer types and magnification levels, stored in a CSV file accessible via Google Drive.

Data Preprocessing

This stage enables data managing, speeding up data processing while minimizing errors. Multiple tasks were performed, with the final one having the focus, which consisted in splitting all images into 3 folders (Train, Validation and Test) which will be fundamental to create a model. The mentioned steps performed were:

Remove Duplicates: Using imagededup library's Percentual Hashing (PHash) method, duplicates were identified and inspected individually – each pair of duplicates appeared twice and in two formats. Firstly, we removed the noise by removing the duplicate rows showing the same duplicate data but in reverse orders and then we identified 14 sequential images, from the same Cancer Class and Type and from the same subject which were substantially impacting dataset integrity and possibly biasing model training, so they were consequently removed.

Handle Missing Data: Looking into the csv file, 4 rows were identified with missing values and filled up with information based on the column path_to_image.

Image Directory: Using path_to_image field in the completed csv file, each image was moved into a single folder 'benign_malignant' for further inspection and management.

First, we proceeded to reorganise the images and split them into 3 folders named train, validation and test. We used stratified sampling to ensure class balance with a 80-10-10% split amongst the folders out of the 7895 images.

For the first stage, we created another two folders named 'Bening' and 'Malignant' inside the 3 folders initially created (Train, Validation and Test).

For the second step, we reorganized the folders again, but this time we split the images into 8 new folders which were named after the cancer types of the images each of them contain.

Over the course of the undersampling stage we replicated the steps performed during the first and second stage but this time the image count was reduced, for the first stage on the malignant side to match the benign count. For the second stage we applied the undersampling to the Ductal Carcinoma cancer in order to reduce it to the same size of the Fibroadenoma cancer which was the cancer type (after Ductal carcinoma) with the most amount of samples but still balanced with the other types. We did this to preserve the information we have on the Ductal Carcinoma as much as possible, but still try to balance it with the other types of cancers.

Image Resize: All images were resized to a standard size, allowing compatibility with the CNN model. In this case, the images were resized maintaining the aspect ratio of the original images, thus ensuring there was no distortion or stretching of image content.

Image Normalization: By normalizing RGB pixel values from [0-255] to [0-1], where 0 represents no intensity and 1 represents full intensity for each color channel, we improved the model's generalization capabilities.

B. Model Selection, Architecture and Training

To develop the CNN model it is important to look to the problem objectives. To ease the path to results, 2 phases were created that share the same backbone model, although with their own differences. Our CNN architecture comprises four convolutional layers, each succeeded by a max-pooling layer. Feature extraction is performed using Conv2D layers with increasing neurons (32, 64, 128 and 128 again), each utilizing a 3x3 filter.

Following each convolutional layer, a MaxPooling2D layer with a 2x2 filter size is applied to downsample the feature maps, thereby reducing spatial dimensions and computational complexity. The extracted features are then flattened into a single vector before passing through a fully connected dense layer with 512 neurons and ReLU activation, which learns higher-level representations and applies non-linearity.

First, we created the train and validation generators based on ImageDataGenerator that has an integrated vectorization feature and allows us to normalize the data. The train and validation generator both serve to define the number of batches, image target size and the class mode. The main difference between train and validation generator lies in the shuffle activation. For training, we apply shuffle=true to force the randomness of the batches, preventing overfitting and counteracting dataset bias. For validation we apply shuffle=False to maintain batch consistency which helps us control and compare validation results.

Secondly, we compiled the information of the model by defining the optimizer, the loss function (binary_crossentropy for the 1st stage and categorical_crossentropy for the second stage) and the metrics which evaluated Recall, Precision, AUC and Accuracy.

Finally the model was trained for up to 10 epochs, with training and validation performance monitored after each epoch using the data generators to manage the large dataset effectively. In addition, for both steps we also relied on the EarlyStopping function to assist us on optimizing the training process by defining the best possible weights and stop running the model as soon as the best val_loss is achieved as per our defined monitoring metric.

Phase A - Train a binary classification model to distinguish between benign and malignant images

Phase B - Extend the model to multi class classification type and predict the images by specific tumor type.

As mentioned, although these classification models differ, they share some similarities that can be leveraged by transfer learning - where the weights of the dense layers were frozen and the final ones were opened to allow the activation function to be changed from one classification model to the other. Some of the most significant differences are:

Data splitting: although the percentages stand the same, the number of folders change in each train, validation and test folders. For binary only 2 are necessary, containing malignant and benign images. On the other hand, the multi class model has 8 folders, one for each type of Cancer.

Activation and Loss Function: Selecting the correct activation and loss functions is crucial for successful neural network training in our binary classification model. For Phase A we chose Sigmoid as activation function in the output layer, which outputs a probability between 0 and 1, hence suitable for this classification model. The Binary Crossentropy loss function was chosen as it effectively measures the difference between the predicted probabilities and the true binary labels. For the multi-class classification model, we used the Softmax activation function, which outputs a probability distribution over all classes. This was paired with the Categorical Crossentropy loss function, which efficiently measures the difference between the predicted probability distribution and the true one-hot encoded labels, optimizing the model for multi-class predictions.

C. Evaluate Model

In order to assess if the model has a good performance, different metrics were analyzed, although one stood out - Recall Metric, as we are dealing with a sensible topic related to medical cancer diagnosis and we need to be sharp classifying the type of cancer. This was used as the north star of the project for its ability to identify all actual positives, for binary classification, and for the multi classification it measured the proportion of actual instances of that class that were identified by the model. In breast

cancer detection, minimizing false negatives is imperative, thus recall was considered the primary evaluation metric to ensure early detection and improve patient outcomes.

Although Recall was considered the most important, other metrics, such as Precision, AUC (Area Under the Curve) and Accuracy were monitored to provide a more holistic comprehensive evaluation of model performance. As optimizer we used Adam, which adapts the learning rate dynamically.

D. Improving Models

In order to improve each model performance several approaches were taken in order to address certain objectives.

Phase A:

For the 1st step we conducted 4 different trials: model1, model2, model3 and model4. On all the trial batch normalization was introduced. For the 1st model, we applied no transformations in order to be able to use it as a control stance. For the 2nd model we applied class weights, data augmentation and callbacks. Here, we also removed a layer of neurons - the last layer with 128 neurons. We applied this with hopes of avoiding model overfitting due to overtraining and we tried to focus on learning more generalized patterns and shapes and not on specific characteristics. We kept this number of layers from here on. For the 3rd model, in addition to the already applied features, we also introduced L2 regularisation and layer dropouts. The 4th model was based on model1, with no applied techniques except early stopping callback. The only performed change was data undersampling to measure the result against the models which contain all the data. These trials were made consecutively to try to understand what works best on improving the model.

Callback: an early stopping callback was implemented to monitor the val_loss metric, preventing overfitting by halting training if no improvement was observed over three consecutive epochs. At first we tried to use val_Recall as our monitoring metric but, unfortunately, as the val_Recall value was very high from the start it was difficult to train the model for more than 3 epochs.

Data Augmentation: balancing the dataset and enhancing model generalization. Techniques included rescale, rotation range, width and height shift, shear rang, zoom range and horizontal flip, ensuring robust training across all magnifications and cancer types.

Class Weights: mitigate class imbalance by assigning higher weights to the underrepresented benign class.

L2 Regularization: different iterations were conducted for each model in order to understand what was the best one.

Layer dropouts: randomly set a fraction of input units to zero during training, preventing overfitting and improving generalization.

Batch normalization: adjust the layers' output in terms of mean and variance, which helps speed up training and improves model stability.

Phase B:

All of the previously mentioned techniques were applied to two models where the only main difference between them relies on the undersampling of data of the 2nd model. This happens because we extend the model with the best results from Phase A (binary classification) to Phase B (multiclass classification).

III. Results

A. Model Phase A

After the previous steps in constructing the models, results were achieved and analyzed. During Phase A four models were created to achieve better results - each model used different approaches:

Model	Attributes
1	Baseline Model
2	Model with Data Augmentation and Class Weights
3	Model 2 with the addition of Dropout and L2 Normalization
4	Equal to the baseline model with data undersampling

In the baseline model, for which we didn't use callback, we trained for 10 epochs, achieving a recall of 82.2% on the test data. However, with the implementation of different attributes, this recall was enhanced to 87.5% for model 2 and it decreased to 87.3% for model 3.

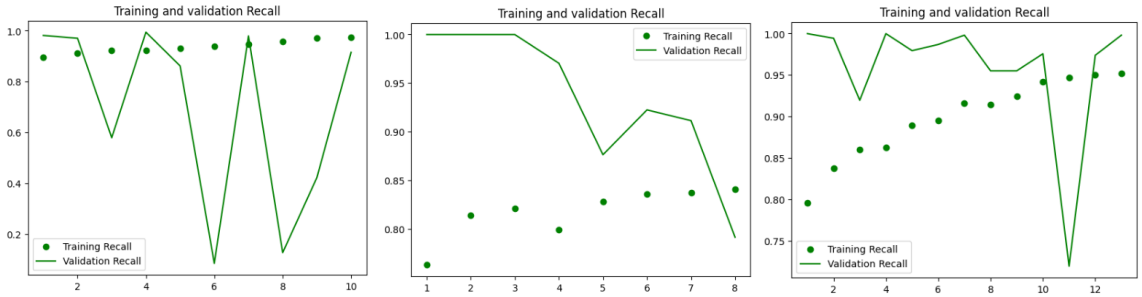

Figure 3 - Model 1, Model 2 and Model 4 ((left to right).

Model 1: The perfect training performance shows the model has memorized the training data but struggles to generalize to the validation set. This explains the dramatic drops in validation recall. Regularization, balancing or having more training data, or simplifying the model architecture could help mitigate overfitting.

Model 2: Shows signs of underfitting combined with potential over-regularization. The slight improvement in training performance and the decline in validation recall indicate the model learns the patterns too slowly and fails to converge optimally. The high initial validation recall compared to training suggests the validation data may be easier, or there could be some issue with the optimization process or data split. Further tuning, such as increasing model complexity or training time might improve this.

Model 4: demonstrates high overall recall, the fluctuations in validation recall suggest instability in its generalization performance. This pattern may indicate sensitivity to noise or insufficient regularization. Model 4, however, still shows generally better recall than Models 1 and 2.

Although the model's evaluation was based on the Recall metric, it should not be looked at alone. As mentioned, other metrics were taken into consideration and we should point out: from model 1 to 3 accuracy falls from 92% to 76% leading to a higher number of Malignant cancers being identified as benign, however precision jumps from 0.1% to 0.81% model 2 and 0.76% in the remaining model. Leading to believe the best overall method is Model 2.

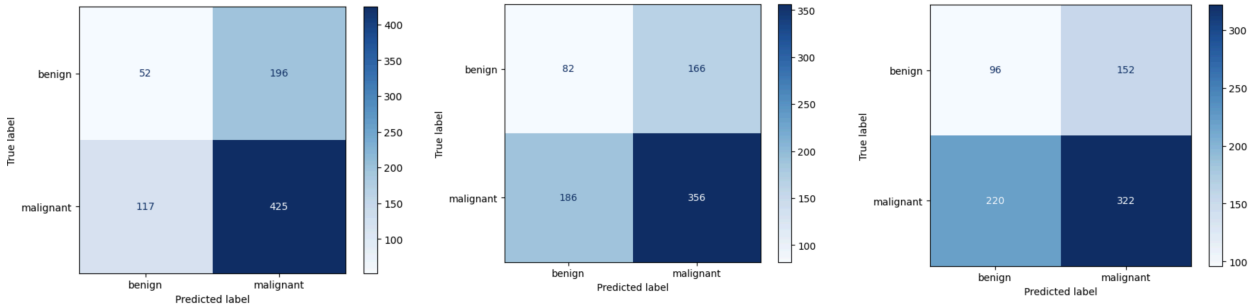

Figure 4 - Accuracy declines from Model 1 to 3 (left to right).

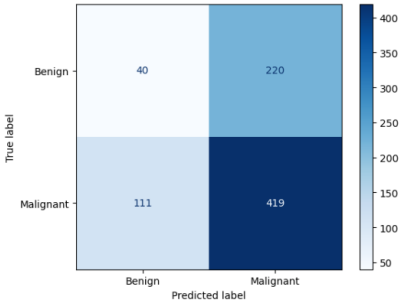

Figure 5 - Accuracy Model 4 (Undersampling).

B. Model Phase B

During Phase B only two models were analyzed, one with baseline parameters and the second an expansion of the first one but with data undersampling. Both used Root Mean Square Propagation (rmsprop) optimizer and were then finetuned using hyperband optimization in order to find the best hyperparameters for the CNN classifier based on Recall.

In this phase Model 2 showed improvements in comparison to Model 1, when comparing the Recall metric - the best performance was obtained after 20 epochs showing values of 69.1% and 60.0%, respectively. When looking into the other metrics accuracy also improved from model 1 to 2 with the best values at 50.0% and 64.5%. Also AUC improved from 86.7% to 91.6%.

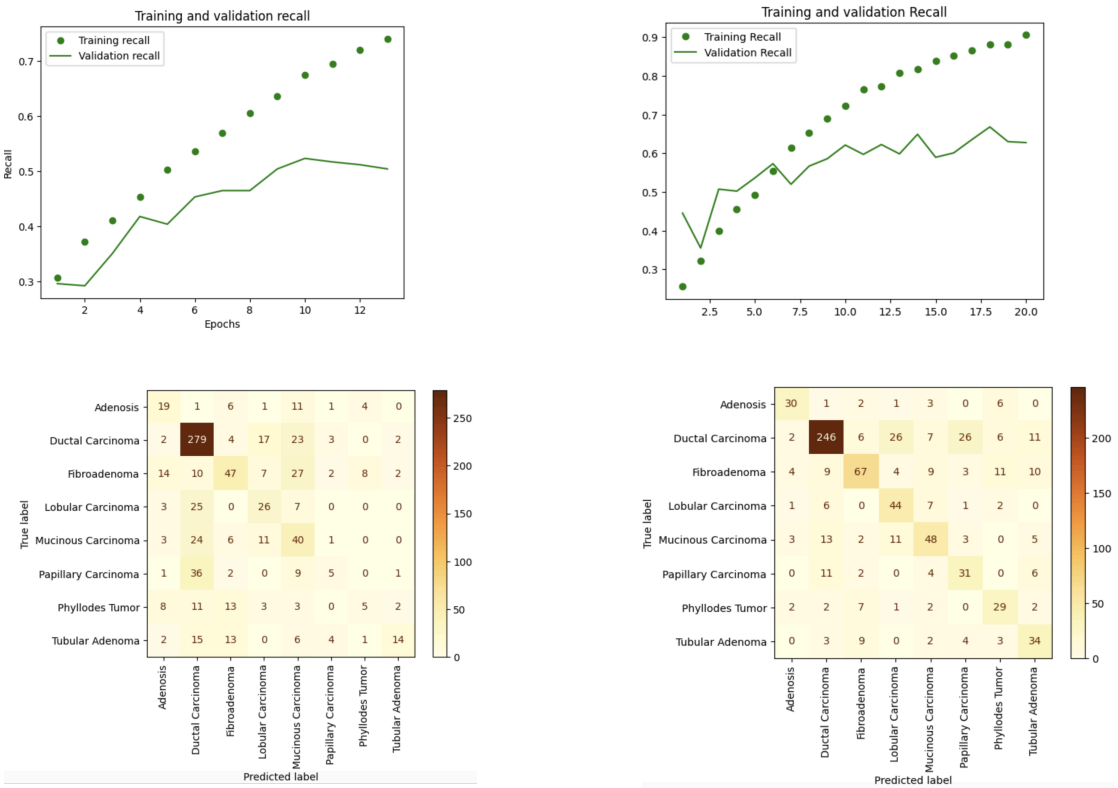

Figure 6 - Progression of Recall (up) metric and accuracy (down) heatmap Model 1 and 2 (left to right).

IV. Conclusions and Future work

The project explored various modeling techniques applied to imbalanced datasets to effectively optimize recall, particularly for critical classes. After extensive experimentation, the models that achieved the best results incorporated a combination of data undersampling and advanced optimization techniques such as class weights (leveraging dynamic weighting to prioritize minority classes), regularization (controlling model complexity to fight overfitting), batch normalization (stabilizing and accelerating the training process), layer dropouts (preventing co-adaptation of neurons for better generalization), data augmentation (introducing variability into training data to improve the ability to generalize). These methods provided significant improvements in recall performance, addressing challenges posed by data imbalance. Models benefited from strong generalization capabilities when implemented with these methodologies. While the project successfully applied techniques to enhance performance, several areas merit further exploration to achieve better results, especially focusing on Stage 1 of the project, which has room for optimization:

Architecture Refinement: experiment with alternative network topologies (e.g., residual networks, attention mechanisms, or Transformers) to better capture patterns within the data.

Test models with varying depths and activation functions to assess their influence on recall performance.

Hyperparameter Optimization: conduct a comprehensive hyperparameter search for aspects such as learning rate, dropout probabilities, and regularization strengths.

Tune class weights for more balanced performance across all classes, especially in severely imbalanced datasets.

Data Handling Improvements: explore hybrid sampling techniques that combine undersampling with synthetic data generation (e.g., SMOTE) to create more representative datasets.

Overfitting Mitigation: regularize further by incorporating techniques like weight decay or early stopping during training.

Smooth out noisy validation metrics through cross-validation to stabilize performance estimates and evaluate model robustness.

Underfitting Resolution: reassess the complexity of the models to ensure they are expressive enough to learn the patterns in the data.

Evaluation Metrics and Testing: evaluate models on a wider range of metrics (e.g., F1-score, precision) to capture a holistic view of performance.

Resources

Breast Center: https://tinyurl.com/msmdphf6
Breast and Economic Benefits of Breast Cancer Interventions: https://tinyurl.com/dtu6y7y5
Alarming rise of late-stage Breast Cancer: https://tinyurl.com/2rkx5wjv
Cancer Stats and Facts: https://tinyurl.com/ymwm33tb
Breast Cancer Facts and Stats: https://tinyurl.com/5hcnvwwv
An Introduction to Convolutional Neural Network (CNN): https://tinyurl.com/3n3e7wcj
PubMed Central: https://tinyurl.com/42yjm7ne

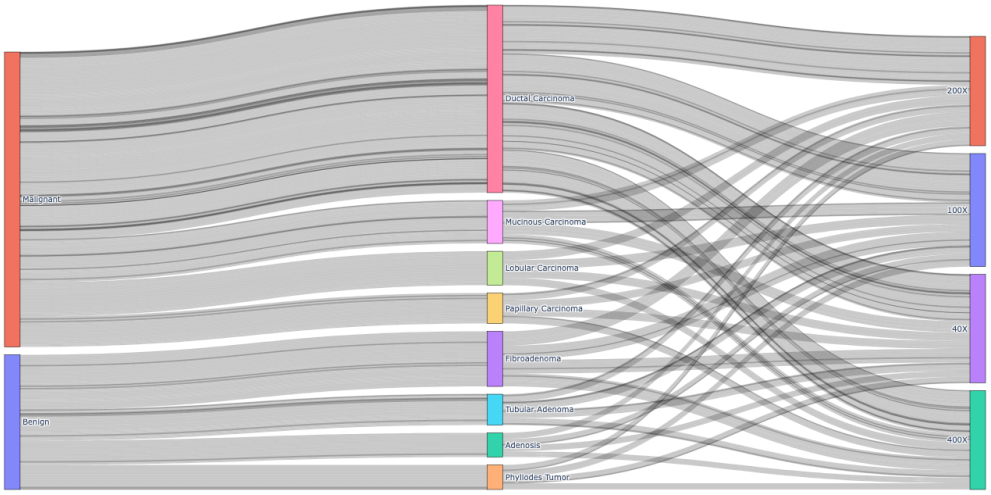

Figure 7 - Number of images per category: Cancer Class, Cancer Type and Magnification (Left to Right).

	path_to_image	Benign or Malignant	Cancer Type	Magnification
2871	BreaKHis_v1/histology_slides/breast/malignant/...	NaN	NaN	NaN
3093	BreaKHis_v1/histology_slides/breast/malignant/...	Malignant	NaN	NaN
3228	BreaKHis_v1/histology_slides/breast/malignant/...	NaN	NaN	NaN
4536	BreaKHis_v1/histology_slides/breast/malignant/...	NaN	NaN	NaN

Figure 8 - Missing values present in the initial csv file.

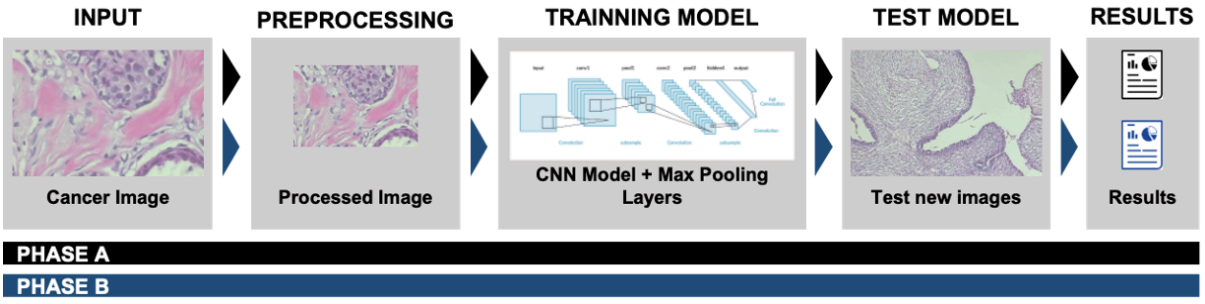

Figure 2 - Model Architecture