EMBED VIEWER

Oriya Sigawy Ayelet Katkov

Advised by Dr. Gil Ben-Artzi

October 19, 2025

Abstract

EMBED (Emory Breast Imaging Dataset) is a large racially diverse mammography dataset, created to address the racial and ethnic lack of diversity in previous datasets.
Researchers are widely using it to develop algorithms to detect and segment cancerous breast lesions.
Despite its importance, accessing and analyzing this dataset can be challenging, as it requires significant coding effort, which limits its immediate usability for many researchers.
To address this issue, we developed an Electron-based GUI that offers a streamlined and intuitive way to explore the EMBED dataset.
The program allows users to filter data by various parameters, view detailed patient information, and save frequently used queries for quick access.
By simplifying dataset accessibility, the tool lowers the entry barrier for researchers and fosters more efficient exploration and utilization of the dataset in medical imaging research.
In addition, the backend of our GUI is an API that gives easier access to the database for data scientists, helps them to understand the data and create detecting models for breast cancer using EMBED.

Contents

1 Introduction

1.1 Breast cancer - background and statistics

Breast cancer is the most commonly diagnosed cancer among women worldwide.
In 2022, about 2.3 million new cases were reported and approximately 670,000 women died from the disease.
Globally, the lifetime risk of developing breast cancer is about 1 in 20 women, while the lifetime risk of dying from it is about 1 in 70.
These numbers vary between countries, depending on health care systems and access to early detection.

In the United States, women are recommended to have annual mammography screening from 40 years to 55 years.
Screening mammograms are recommended for asymptomatic patients to detect occult breast cancer.
Approximately 90% of screening mammograms are normal, but approximately 10% of screening exams demonstrate an abnormality that requires further imaging.

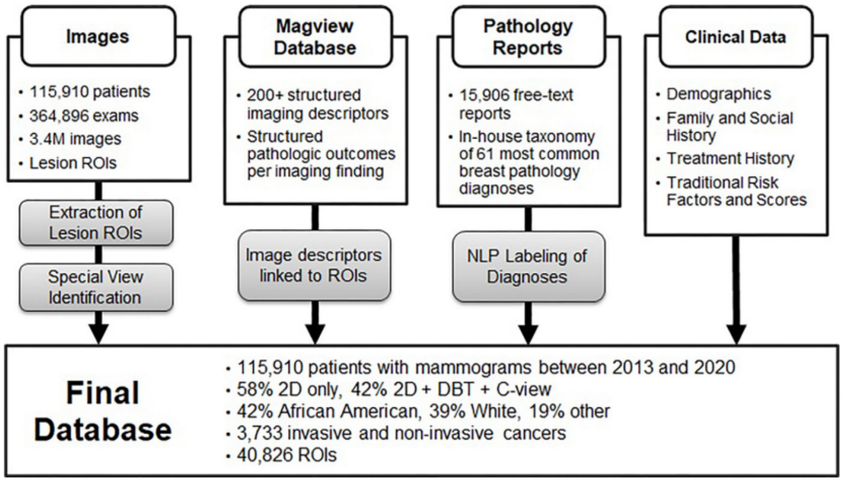
Detecting breast cancer early is one of the most effective ways to save lives.
When the disease is found at an early stage, treatment is generally less invasive, more successful, and survival rates are much higher.
For example, in the United States, the five-year survival rate for women diagnosed with localized breast cancer (cancer that has not spread outside the breast) is about 99%, compared with only 31% when the cancer is found at a late, metastatic stage.
Early detection also reduces treatment costs and side effects, improving quality of life.

Unfortunately, many women are still diagnosed at later stages, often because of limited access to regular screening or because early signs are missed.
This highlights the need for better tools, such as deep learning models, to improve accuracy in breast image analysis and to support radiologists in finding cancers earlier.

Developing such models and predictable tools require large, high-quality datasets of breast images together with clinical and pathologic information.
Without diverse and well-annotated datasets, AI systems risk being less accurate, especially for underrepresented patient groups.

1.2 Existing datasets

Unfortunately, most publicly available breast cancer datasets have several important limitations:
They are often relatively small in size, include little or no information such as image annotations or pathology results, and most critically, they tend to be ethnically and racially homogeneous.
This lack of diversity is a major concern. In particular, African American women and other minority groups are consistently underrepresented in breast imaging datasets as well as in many other health care–related datasets, even though these groups face disproportionately poor breast cancer outcomes.
For example, a recent large-scale study by Google showed that using artificial intelligence models together with radiologists could improve recall rates in breast cancer screening.
However, the study included very few African American patients, limiting the ability to understand how well the model performs in this population.
This highlights a broader problem:



Images
• 115,910 patients
• 364,896 exams
• 3.4M images
• Lesion ROIs
Extraction of Lesion ROIs
Special View Identification
Magview Database
• 200+ structured imaging descriptors
• Structured pathologic outcomes per imaging finding
Image descriptors linked to ROIs
Pathology Reports
• 15,906 free-text reports
• In-house taxonomy of 61 most common breast pathology diagnoses
NLP Labeling of Diagnoses
Clinical Data
• Demographics
• Family and Social History
• Treatment History
• Traditional Risk Factors and Scores
Final Database
• 115,910 patients with mammograms between 2013 and 2020
• 58% 2D only, 42% 2D + DBT + C-view
• 42% African American, 39% White, 19% other
• 3,733 invasive and non-invasive cancers
• 40,826 ROIs

Figure 1: EMBED - Architecture

when datasets lack diversity, research findings and technologies developed from them may not generalize equally across all patient groups.
Training an effective deep learning model required a large, labeled, and diverse datasets to ensure generalizability.
The Emory Breast Imaging Dataset came to achieve this goal, and given that 45% of the patients at the Emory institution are African American, they were able to curate a diverse dataset to represent African American women in breast imaging research.

1.3 Overview of the Emory Breast Imaging Dataset

EMBED contains lesion-level annotations, pathologic outcomes, and demographic information for 116000 patients from racially diverse backgrounds and will help bridge the existing gaps in granularity, diversity, and scale in breast imaging datasets.

The full version of the EMBED dataset includes about 3.38 million screening and diagnostic mammographic images, 364,000 exams of approximately 110,000 patients from four hospitals over an 8-year period (January 2013 to December 2020).

The data has been combined from four Emory institutional hospitals - two community hospitals, one large inner-city hospital, and one private academic hospital.

2 Methodology

2.1 EMBED - Construction Overview

You can see the architecture of EMBED in Figure 1

2.2 EMBED - Statistics

You can see the statistics of EMBED in Figure 2

2.3 EMBED AWS Open Data release

We worked with the EMBED AWS Open Data release,
which represents 20% of the dataset divided into two equal cohorts at the patient level. This release of the dataset includes 2D and C-view images.

Statistics:

- Patients: 22,382

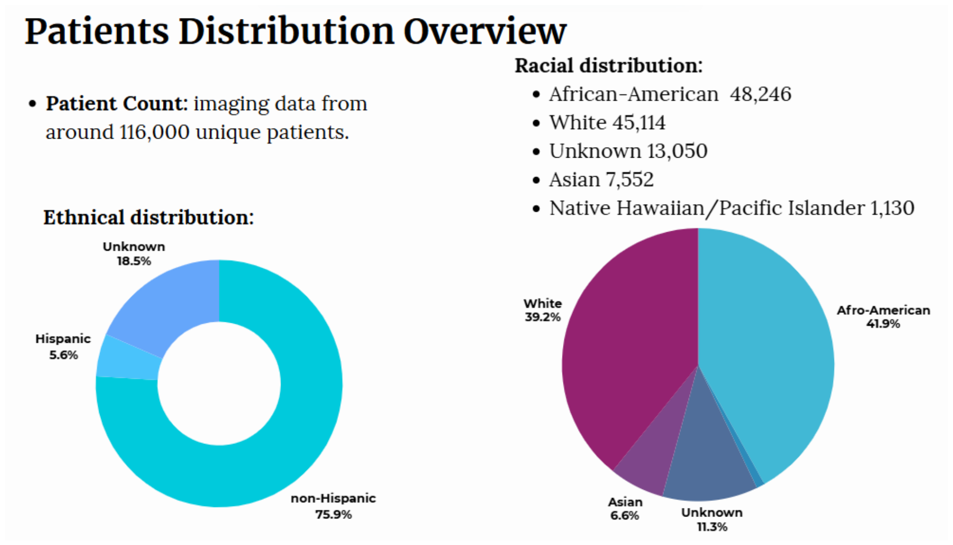


Figure 2: Statistics of EMBED patients

- Exams: 76,373 (72% screening, 28% diagnostic)
- Images: 480,323 (76% 2D, 24% C-view)

File Structure:

- cohort → patient → exam → image
- No patient or exam overlap between cohorts

Data Tables:

- Clinical Data Legend (AWS_Open_Data_Clinical_Legend.csv)
- Clinical Data (EMBED_OpenData_clinical.csv)
- Metadata (EMBED_OpenData_metadata.csv)
- Reduced Versions (Simplified tables for easier analysis)

2.3.1 Tables description

- Metadata (EMBED_OpenData_metadata.csv):
  - One row per image, 165 columns.
  - Indexed by image with exam and file-level information.
  - Key fields: anonymized patient and exam IDs, anonymized exam date, DICOM paths, PNG paths, study details, image type, breast side, mammography type, spot_mag, number of ROIs and ROI coordinates.
- Clinical Data (EMBED_OpenData_clinical.csv):
  - One row per finding, 112 columns Findings per exam recorded with imaging and pathology details.
  - Pathology manually linked to findings.
  - Key fields: anonymized patient and exam IDs,anonymized exam date, patient race, patient ethnicity study description, tissue density, BIRADS scores, breast side, biopsy type, mass/calcification descriptors, pathology outcomes.

We will expend further on ROIs and on the mammography type, spot_mag fields.

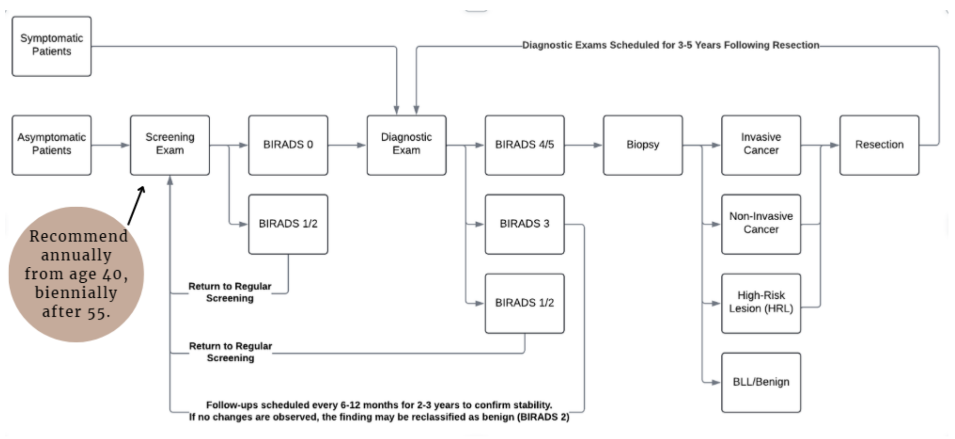

Symptomatic Patients
Diagnostic Exams Scheduled for 3-5 Years Following Resection
Asymptomatic Patients
Screening Exam
BIRADS 0
Diagnostic Exam
BIRADS 4/5
Biopsy
Invasive Cancer
Resection
BIRADS 1/2
BIRADS 3
Non-Invasive Cancer
Recommend annually from age 40, biennially after 55.
Return to Regular Screening
BIRADS 1/2
High-Risk Lesion (HRL)
Return to Regular Screening
BLL/Benign
Follow-ups scheduled every 6-12 months for 2-3 years to confirm stability.
If no changes are observed, the finding may be reclassified as benign (BIRADS 2)

Figure 3: analyzing process

2.4 Screening and analyzing process

The process depicted in Figure 3.
Asymptomatic patient starts with a screening exam (recommended annually for women from age 40 and biennially from age 55).
The result of screening exam can be BIRADS 0, 1 or 2:

- If the result is 1 or 2 it means everything is normal.
- If the result is 0, the patient will do a diagnostic exam.

The result diagnostic this exam can be BIRADS 1, 2, 3, 4, 5:

- If the result is 1 or 2 it means everything is normal.
- If the result is 3, it means the patient needs follow-up exams every 6-12 months for 2-3 years.
- If the result is 4 or 5, the patient will do a biopsy that will determine whether the cancer is benign or not. If the cancer is not benign, the patient will have to do a resection.

Summarize the BIRADS results:

- BIRADS 0 - require further imaging (diagnostic)
- BIRADS 1 - negative, no cancer
- BIRADS 2 - benign
- BIRADS 3 - need short-term follow-up (6 months)
- BIRADS 4 - suspicious (proceed to biopsy)
- BIRADS 5 - highly suspicious (proceed to biopsy)
- BIRADS 6 - known biopsy (proven malignant cancer)

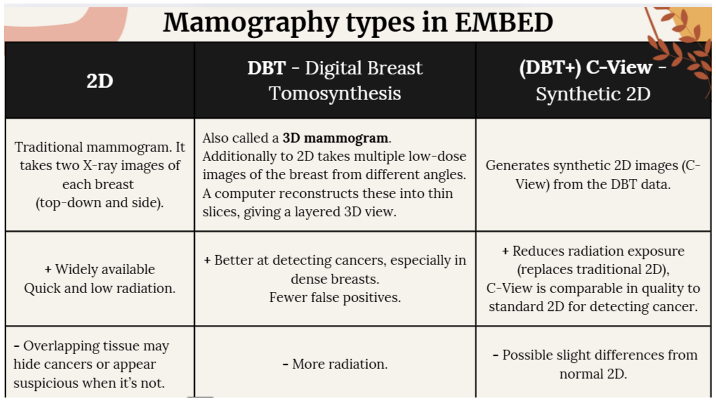
2.5 Mammography types

You can see the mammography types in Figure 4
spot_mag - Indicates if the image is a special view such as spot compression or magnification You can see the view types and differences in Table 1

2.6 ROIs - regions of interest

ROIs are collected by mapping radiologist-annotated images back to their original source images (primary to secondary match).
Each ROI is represented as a bounding box: [ymin, xmin, ymax, xmax].
You can see example for that in Figure 5.
Extraction algorithm of lesion ROIs (available for 2D and synthetic 2D):

1. Starting with burnt in annotated ROI low res image



Mamography types in EMBED

2D	DBT - Digital Breast Tomosynthesis	(DBT+) C-View - Synthetic 2D
Traditional mammogram. It takes two X-ray images of each breast (top-down and side).	Also called a **3D mammogram**. Additionally to 2D takes multiple low-dose images of the breast from different angles. A computer reconstructs these into thin slices, giving a layered 3D view.	Generates synthetic 2D images (C-View) from the DBT data.
+ Widely available Quick and low radiation.	+ Better at detecting cancers, especially in dense breasts. Fewer false positives.	+ Reduces radiation exposure (replaces traditional 2D), C-View is comparable in quality to standard 2D for detecting cancer.
- Overlapping tissue may hide cancers or appear suspicious when it's not.	- More radiation.	- Possible slight differences from normal 2D.

Figure 4: Mammography types

View type	Purpose	How it's done
Spot Compression	Flatten a small area to "spread apart" tissue	Apply a smaller paddle over the region of interest
Magnification	Visualize microcalcifications in greater detail	Increase OID (object-image distance) and use a small focal spot; usually CC or MLO orientation but zoomed in
Lateral/ML	Localize a lesion in the medial/lateral plane	90° view from side (mediolateral or lateromedial)

Table 1: view types

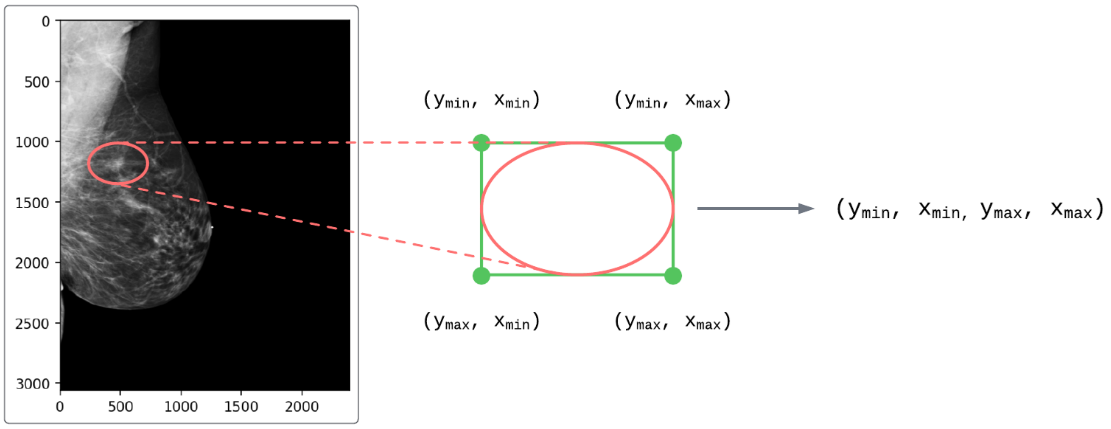


Figure 5: ROI example

2. Detecting ROI Bounding Box coordinates via Deep Learning [acc 99.99% mIoU 0.95]
3. Finding matching original mammogram using Image Similarity [python]
4. Saving a crop of the original mammogram according to ROI coordinates

~0.98 accuracy

Metadata of the ROI:

- **num_roi:** Number of ROIs in an image
- **DCM_ROI_coords / PNG_ROI_coords:** ROI coordinates for DICOM and PNG images, stored as nested lists.
- **ROI_match_level:** A nested list encoding match levels (1 = primary, 2 = secondary).
- **ROI_source:** Indicates the original image an ROI came from.
- **SSC_ROI_flipped / SSC_ROI_dest / SSC_match_level:** These columns track orientation and mapping between 2D and C-View images.
- **PNG_flipped:** Indicates if a PNG was flipped during DICOM to PNG conversion.

3 EMBED viewer

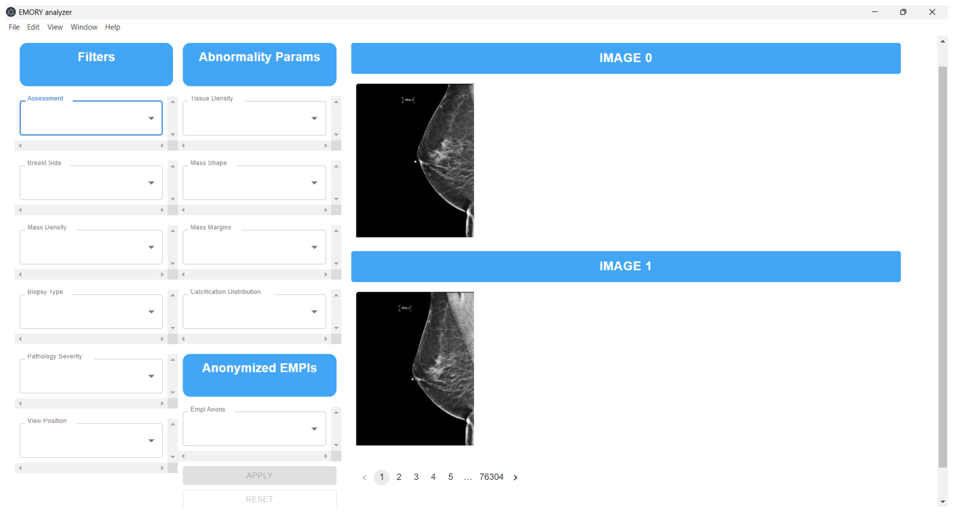


Figure 6: EMBED viewer

The viewer has three components:

- **Backend** – responsible for handling requests, retrieving information, formatting it, and returning the results.
- **Frontend** – responsible for presenting the viewer, including the filter options, the images, and the image data.
- **Analyzer** – runs both the backend and the frontend.

3.1 Backend

This component is written in Python and uses the Flask library to handle data requests and processing. The backend receives patient data and images, along with related information from CSV files and AWS (via an S3 connection).
It merges the patient data with the image data so that each image is associated with its corresponding patient.
It also downloads images from S3 and uses a caching mechanism for efficiency.
The backend provides 9 endpoints that allow users to:

- Retrieve the available dataset filters and their options
- Retrieve all anonymized Enterprise Master Patient Index (EMPI) entries in the dataset
- Retrieve all image IDs (an internal index created for easier communication between the backend and frontend)
- Retrieve a PNG file of an image (using the image ID)
- Retrieve all metadata of an image (using the image ID)
- Retrieve all image IDs that satisfy specific conditions (filters)

The backend can be used as a server not only for our frontend but for everyone that wants to access the EMBED dataset.
Anyone can examine and test the database through the backend endpoints.

It is also very useful for model training - through our backend you can download a bunch of images that satisfy specific conditions and define your features by that, without digging inside the database itself.

3.2 Frontend

This component is implemented as an Electron-based desktop viewer, designed to provide a GUI for interacting with the dataset.

It is developed primarily in TypeScript and JavaScript, while also incorporating standard web technologies such as HTML and CSS to build the UI.

The communication between the frontend and backend is facilitated through Electron's Inter-Process Communication (IPC) mechanism.
Specifically, the frontend issues requests via the IPC agent, which internally leverages Node.js modules to transmit HTTP requests to the backend.

This architecture ensures efficient data exchange between the two components while maintaining a clear separation of responsibilities.

When the viewer is launched, the frontend initiates a sequence of requests to the backend.
First, it retrieves the set of available filters together with their corresponding options.
These filters allow the user to refine and structure the dataset according to specific parameters.
Following this, the frontend requests a small subset of the data—namely, the first two images in the dataset along with their associated metadata—which are presented to the user on the opening screen as an introductory view of the system's capabilities.

The viewer's frontend provides three core functionalities:

1. **Applying filters** – The system allows users to apply a range of filter conditions in order to query the dataset.
   You can see an example of a filter in Figure 7.
   Once filters are defined, the frontend transmits these conditions to the backend, which processes them and returns a list of image IDs that meet the specified criteria.
   The frontend can then request the corresponding images by referencing their IDs.

   The filters you can filter on:

   - **Anonymized EMPI** - you can filter by specific anonymized Enterprise Master Patient Index, i.e. getting all the images of specific patient
   - **Assessment -** you can filter by the BIRADS value of the image
   - **Breast side**
   - **Mass density** - you can filter by how dense a breast lesion (mass)
   - **Abnormality type** - you can filter by what kind of suspicious or notable feature is present in the breast
   - **Pathology -** you can filter by the pathology result of the image
   - **View position** - you can filter by type of view acquired, such as CC or MLO

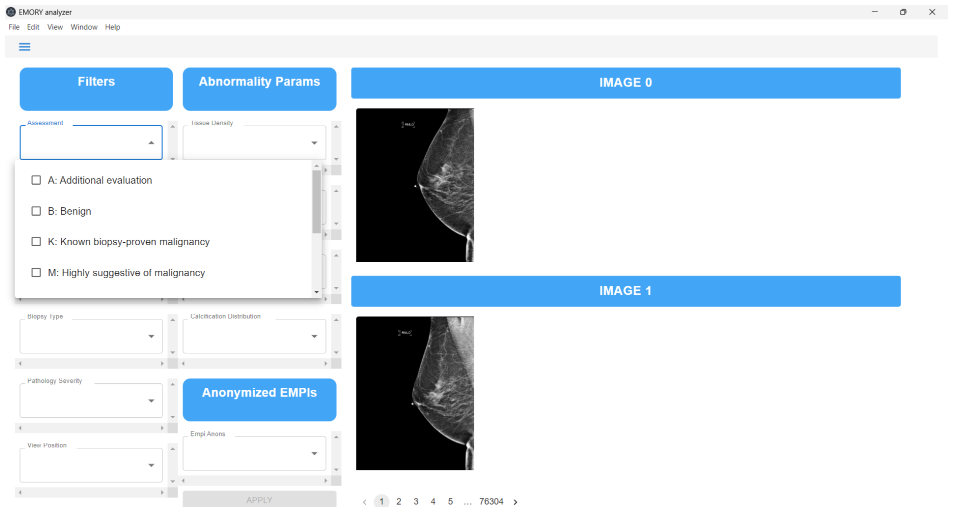


Figure 7: Viewer with option to Assessment field

- **Tissue density** - you can filter by how much fibroglandular (dense) tissue compared to fat is present in the breast or in a specific lesion.
- **Mass shape** - you can filter by overall outline or form of a lesion
- **Mass margins** - you can filter by the form of the edge or border of a breast lesion
- **Calcification distribution** - you can filter by how calcium deposits (tiny white spots) are spread within the breast tissue.

Figure 7 is the viewer with the options for the assessment filter.

2. **Saving queries** – To improve usability and efficiency, the viewer supports the saving of frequently used queries.
You can see how to do that in Figure 8.
This feature enables users to store and reuse filtering conditions without needing to re-enter them manually, thereby streamlining repetitive analytical tasks.
To save your query, select filters, apply them and then click on "save query".
you will have to give this query a name, and then by clicking in the menu in the upper-right corner you will get your saved queries.
Then you can apply or delete them (See figure 8).

3. **Presenting patient metadata** – In addition to retrieving images, the viewer also provides access to detailed metadata associated with each image.
This metadata offers contextual information that supports further analysis and interpretation by the user.
If in the dataset the value of a field in the metadata is null, this field will not be presented.
You can see example for that in Figure 9.
The metadata includes:

   - Image ID
   - Anonymized Enterprise Master Patient Index (EMPI)
   - Anonymized exam ID
   - BIRADS breast density
   - The BIRADS score of the exam
   - The breast side
   - Mass shape according to BIRADS descriptors.
   - Mass margin according to BIRADS descriptors.
   - Mass density according to BIRADS descriptors.
   - Distribution of calcifications according to BIRADS descriptors.
   - How the tissue sample was collected (e.g., biopsy, FNA, lumpectomy)

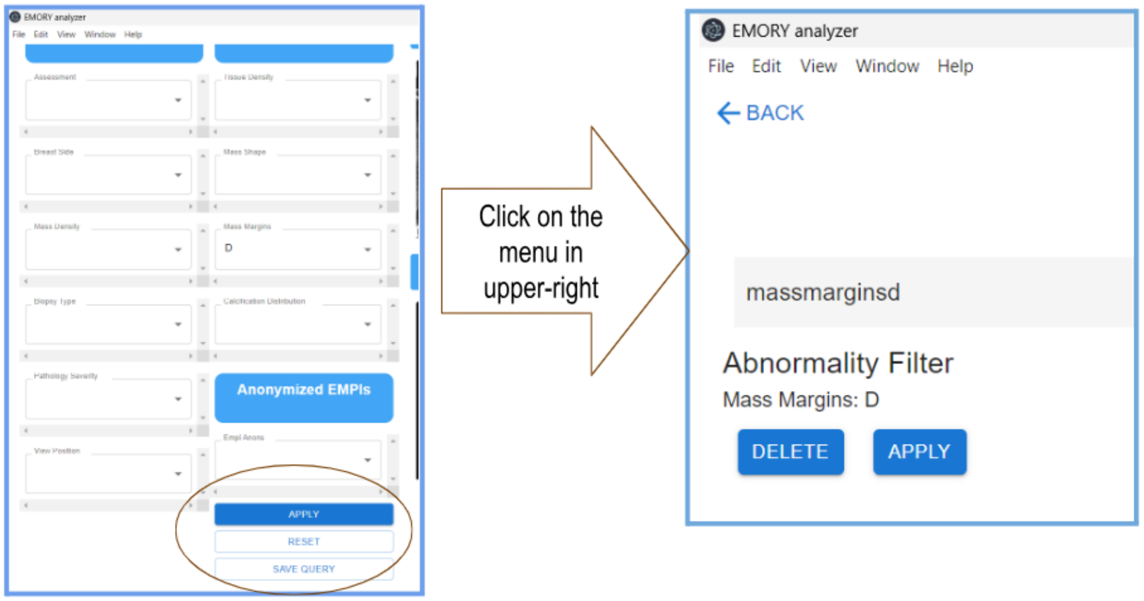


Figure 8: Save Query Process

- The most severe pathology result from a given specimen
- Type of view acquired such as CC or MLO
- Number of ROIs for a given file

3.3 Analyzer

This component is designed to streamline the deployment and execution of both the backend and the frontend systems.

Instead of requiring users to manually clone and configure these two projects separately, the component automatically handles this process.
It ensures that both repositories are present locally and verifies that the latest versions are being used by checking their corresponding Git sources.
If updates are available, the component pulls the most recent changes, guaranteeing that the environment remains synchronized with the most current codebase.

To facilitate execution across different operating systems, the analyzer includes platform-specific scripts.
For **Windows** environments, the **runnable.ps1** script is provided, enabling automated initialization of both the frontend and backend services.
Similarly, for **Linux** and **macOS** systems, the **runnable.sh** script performs the same function.

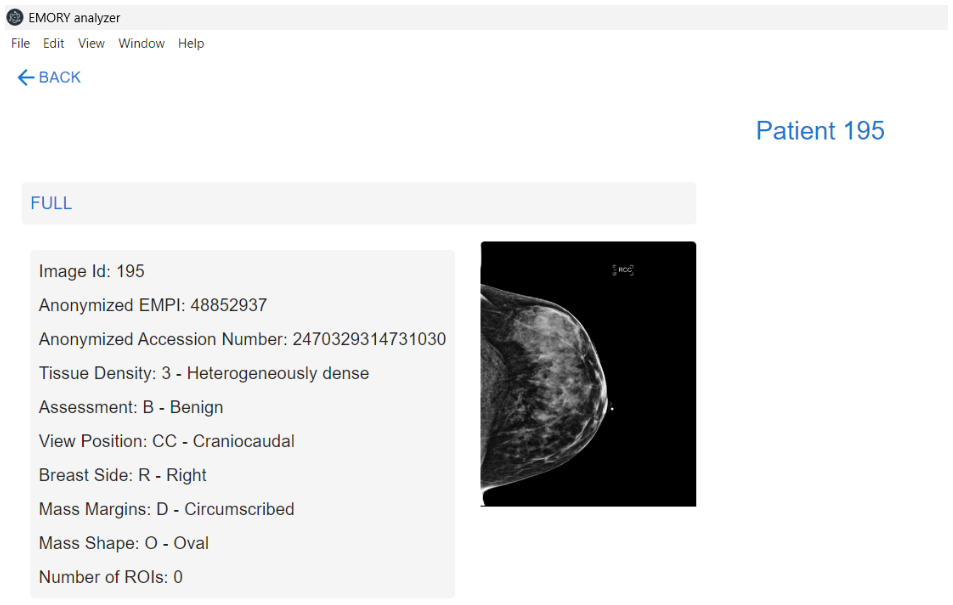
4 Discussion

4.1 Contributions

Our viewer provides a practical solution for accessing and analyzing the EMBED, which is a large-scale breast cancer dataset, stored in AWS (which also makes the access more difficult).

By offering a user-friendly interface, the system reduces the technical barriers associated with managing raw data files, enabling clinicians and researchers to efficiently access medical images alongside their corresponding clinical and pathological information.

This functionality is particularly valuable in clinical settings, where timely access to both imaging and associated metadata can support decision-making and research efforts.
From a research perspective, the viewer plays a crucial role in supporting the development of AI and deep learning models for breast cancer detection and classification.



Patient 195

Image Id: 195
Anonymized EMPI: 48852937
Anonymized Accession Number: 2470329314731030
Tissue Density: 3 - Heterogeneously dense
Assessment: B - Benign
View Position: CC - Craniocaudal
Breast Side: R - Right
Mass Margins: D - Circumscribed
Mass Shape: O - Oval
Number of ROIs: 0

Figure 9: Image with metadata

By enabling data scientists to directly explore, visualize, and interpret the dataset, the viewer provides an essential foundation for understanding data distributions, identifying relevant features, and selecting appropriate labels for predictive modeling.
This step is critical for building robust and clinically meaningful algorithms that can assist in early cancer detection, differentiation between benign and malignant findings, and even subtype classification.

In addition, the viewer's backend operates as a fully functional API, offering a programmatic interface to the dataset.
Researchers can query the dataset through HTTP requests to obtain statistical summaries, retrieve metadata on imaging studies, or download subsets of images that meet predefined conditions (e.g., lesion type, patient demographics, or imaging modality).
This capability significantly accelerates the process of preparing training and test sets for machine learning workflows, making the dataset not only more accessible but also more customizable for diverse research needs.

Overall, the viewer bridges the gap between raw data storage and actionable medical research.
It supports clinicians by simplifying access to complex datasets, and it empowers data scientists by providing the tools needed to construct and evaluate predictive models.
In doing so, it contributes to the broader goal of advancing breast cancer research, with the potential to improve diagnostic accuracy and patient outcomes.

4.2 Limitations

Our viewer have several limitations:

First, the initial loading time of the viewer is relatively slow, because it establishes a connection with AWS, and the backend needs to organize the very large dataset.
It takes time for it to give the frontend the images and data for the opening screen.

Second, the process of downloading and rendering the images is resource-intensive.
Each image requires time to download from AWS and display within the viewer.
This can reduce efficiency when browsing large numbers of images consecutively, especially in research workflows that demand high-throughput data exploration.

Finally, storing images locally introduces additional constraints.
Downloaded images take lots of disk space on the user's computer.
This can create storage limitations.

5 Experiments & Models

The second part of our project, after the development of the viewer, includes research, and deep learning models execution on different aspects of the EMBED dataset.

Below are presented results of different models achievements on several classification tasks. For each task fitting mammography images were downloaded and grouped by necessary criterions of their metadata or clinical information, e.g. BIRADS assessment, or pathological findings results' severity.

Preprocessing:

Since the images in the dataset are not of equal forms or size, and many pytorch's models demand input images to be of equal size, before running any of the models whose results are about to be presented all of the images were resized to 256x256 pixels, and then center cropped to 224x224 pixels (the important potentially cancerous areas aren't in the edges so no harm was caused.)

Data split is constant 80% for training and 20% for testing.

Results metrics:

In most of the following reported experiments the models' performance will be measured in two metrics, accuracy and ROC-AUC (Area Under the Receiver Operating Characteristic Curve) since the classification of cancer and pathological results isn't morally same weighted as benign and negative findings, it is as important for us to check consistency in higher scores to the correct class and not only percentage of correct final predictions out of all predictions.

Classification fields:

The metadata and the clinical data files indeed contain various fields and information about the images and exam findings respectively, out of them the main fields that caught our attention as fascinating to research DL models' ability to classify and predict accurately as a tool to potentially help researches and medical personal were the BIRADS assessment of each image, and later on the pathology severity field.

5.1 Experiment 1 "2b6k"

Reminder - BIRDADS assessment per mammography is a scale metric indicating the assessed health (breast cancer wise) of the breast tested as presented below.

- BIRADS 0: A – Additional evaluation
- BIRADS 1: N – Negative
- BIRADS 2: B - Benign
- BIRADS 3: P – Probably benign
- BIRADS 4: S – Suspicious
- BIRADS 5: M- Highly suggestive of malignancy
- BIRADS 6: K - Known biopsy proven

The first trial experiment is simply a binary classification of 100 images with BIRADS value 2 vs. 100 images with BIRADS value of 6.

These specific classes were chosen for the initial trial executions due to their relative certainty - either certainly benign findings, or certain biopsy proven malignancy.

For this first experiment two models were employed:

1. CovNeXt tiny (2022) - pytorchs convnext tiny model, a pure convolutional model structured with inspiration from visual transformers. We used the pretrained ImageNet weights.
2. ResNet18 (2015) - a residual gradient convolutional neural network, 18 layers version. We used the pretrained ImageNet weights here too.

Table 2 below displays the performance of the models described above.

Each cell contains the accuracy as the upper value, and bottom roc-auc value. Bold values indicate the best result for each model.

100 2B vs. 100 6K	ConvNeXt Tiny	ResNet18
5 epochs	90 0.9349	87.5 0.9449
10 epochs	**95** **0.985**	90 0.913
15 epochs	95 0.985	**92.50*** 0.9635
20 epochs	87.5 0.9105	

Table 2: Performance comparison of ConvNeXt Tiny and ResNet18 across different epochs. The upper number indicates accuracy (%), and the lower number indicates AUC.

Note: From here an continuing in next experiments, once performance started dropping in a given number of epochs, from our experience so far it didn't raise in larger number of epochs, an overfitting process took place probably, therefore once performance dropped significantly e.g. in 15 epochs, the following cells (20 epochs) are empty, it was useless to continue, peak result was reached.

Conclusions:
As we can see in table 2 the models handle very well the task of classifying mammographic images solely between the "benign" and the "known biopsy proven" malignant classes.

So far the best performer is the ConvNext tiny model, reaching at its peak an accuracy of 95% and a roc-auc value of 0.985, showing outstanding consistency in class separation.

5.2 Experiment 2 "All 7 BIRADS"

Now, seeing that the models can distinguish mammographic images between BIRADS dominant assessments, we would like to take a step forward and look how will the models handle a more real-world issue of predicting a full BIRADS assessment, with different outcomes available, taking a shot at real medical process and look whether models can try to compete with human performance.

For this second experiment three models were employed:

1. ConvNext tiny - same as in previous experiment.
2. ResNet18 - same as in previous experiment
3. ResNet34 (2015) - a model with similar approach as ResNet18, but deep with more residual blocks, 34 layers. The deeper network can potentially enable better feature extraction and more attention to important nuances in mammograms. Therefore, we expect to see a slight improvement in results compared to ResNet18, trading it for slower computation.

Conclusions:
We can see some interesting points from table 3 , first of all surprisingly ResNet18 outperformed ResNet34.

Second of all, results reached 65.71% accuracy (achieved by ResNet18), which regarding classification between 7 labels is a nice result, with matching 0.9123 roc-auc value, showing very good consistency in class separation, which is also very important with different BIRADS categories.

5.3 Experiment 3 "Diagnostic 456"

For the next experiment the focus and the classification target will be the BIRADS values that can appear after a diagnostic examination, specifically BIRADS 4 (suspicious), 5 (highly suggestive of malignancy), and 6 (known biopsy proven malignancy).

The first attempt includes a ResNet34 model similarly to previous experiments.

All 7 BI-RADS	**ConvNeXt Tiny (ImageNet)**	**ResNet18 (ImageNet)**	**ResNet34 (ImageNet)**
5 epochs	51.67 0.8683	55.00 0.8590	44.29* 0.8485
10 epochs	**59.17** 0.8857	**65.71*** 0.9123	55.71* 0.8769
15 epochs	56.67 0.8132	62.14* 0.8983	**59.29*** 0.8849
20 epochs			56.43* 0.8792

Table 3: Performance of different models on all 7 BI-RADS categories across multiple epochs. The upper number indicates accuracy (%), and the lower number indicates AUC.

Diagnostic456	**ResNet34 (ImageNet)**
5 epochs	**88.04** **0.9457**
10 epochs	88.04 0.9430
15 epochs	86.96 0.9332

Table 4: Performance of ResNet34 (ImageNet) on the diagnostic BIRADS 4 5 and 6 categories images across multiple epochs. The upper number indicates accuracy (%), and the lower number indicates AUC.

See results in table 4. As previously, upper value - accuracy, lower value - roc-auc.
Table 4 shows us best performance at 5 epochs' reaching 88.04% accuracy and 0.9457 roc-auc. Very nice results for a triple class classification.

Following the instruction of our supervisor Dr. Gil Ben-Artzi we tried to search for a scientific paper with linked code or github or resource with published model weights or saved state model of models like resnet18 or convnext tiny or resnet 34 or other, after training on medical relevant data, ideally mammograms and breast cancer datasets.
Surprisingly nothing fitting the description was found at the time, except for a few exceptions, one of which was the following github repository by Eric A. Scuccimarra, which includes multiple built models, trained on a combination of DDSM dataset and abnormalities ROI extracted from CBIS-DDSM. The repository includes the code of the models, and saved states with saved weights after training. Our deep gratitude goes to Mr. Scuccimarra for the neatly organized and open information, code, and models.

Since the models were saved with TensorFlow v1, which couldn't use the saved state directly, but we did the next best thing we could think of - create a model with the exact same layers with pytorch and then reapplying the saved state weights to this model.
You can check out the structure of the reconstructed model layers in the end of the project book, in the Attachments.
As a part of the experiment we left a different number of last layers unfrozen, and fine-tuned them to the EMBED dataset.
See results in table 5.
Conclusions:

Num. of unfrozen last layers	**Performance**
1 unfrozen last layer	30 0.5078
2 unfrozen layers	40 0.6135
3 unfrozen layers	25 0.5352

Table 5: Performance of reconstructed "model_s1.0.0.29l.8.2" across different fine-tuning depths.

The result don't look assuring, since 33% for 3 categories classification is just random selection, so the model reaching at most 40% is very low.Perhaps there is quite a difference between the classification aims between DDSM and EMBED.

Future work to improve this may include adapted normalization, bigger dataset, more training, trying other frozen layers combinations, etc.

5.4 Experiment 4 "Pathology Severity"

Stepping up, there is another interesting field in the dataset, the "path_severity" label in the clinical data .csv file.

As explained in the EMBED Github repository's readme file, it denotes "The most severe pathology result from a given specimen".

The values this field can receive are:

- 0 : invasive cancer
- 1 : non-invasive cancer
- 2 : high-risk lesion
- 3 : borderline lesion
- 4 : benign findings
- 5 : negative (normal breast tissue)
- 6 : non-breast cancer

Getting a deep learning model to predict correctly the pathological severity of a given mammogram, without any information regarding the biopsy or clinical analysis is an extremely fascinating mission.

The forth experiment attempts a naive approach, directly predicting the pathology severity label given the mammogram, comparing different models' performance on this task, the fifth and final experiment will be an attempt a more tricky information combination.

Note: An important step that significantly improved performance was computing the mean and standard deviation of the combined images across all pathology severity classes (200 images times 6 classes = 1200 images total, so pretty representing the dataset), and normalizing the images before running them through the models.

The mean was 0.1979 and the std 0.1969 (pixels scaled to [0 ,1]), originally 50.46, 50.21 for pixels [0, 255].

Table 6 demonstrates the advantages of using a personalized normalization. Amazing to see, 12%, 14%, 9% just in accuracy. Impressive impact.

Table 7 summarizes the accuracy and roc-auc of convnext tiny, resnet18 and resnet34 on the pathology severity sub-dataset:

Conclusions:

Firstly, personalized normalization has a big impact on the accuracy of the models. Secondly, in this experiment the best result was held by ResNet34, reaching 56.17% accuracy and 0.8476 roc-auc, which is a mid-nice result to a 6 class classification.

path_severity, 5 epochs	**ConvNext tiny (imagenet)**	**ResNet18 (imagenet)**	**ResNet34 (imagenet)**
before normalization	36.16 0.7321	38.72 0.7531	41.70 0.787
with normalization	**48.09*** 0.7968	**54.04*** 0.8297	**50.64*** 0.8011

Table 6: Normalization's impact on performance, tested on `path_severity` field images (5 epochs).

path_severity	**ResNet34 (imagenet)**	**ResNet18 (imagenet)**	**ConvNext tiny (imagenet)**
before training (base)	50.64* 0.8011	54.04* 0.8297	48.09* 0.7968
5 epochs	**56.17*** **0.8476**	50.64* 0.8234	47.23* 0.8125
10 epochs	51.06 0.8232	52.77* 0.8227	48.51* 0.8013
15 epochs			

Table 7: Performance comparison of ResNet34, ResNet18, and ConvNext tiny (ImageNet pretrained) on `path_severity` dataset across different training epochs.

5.5 Experiment 5 "ViT BIRADS"

This experiment might be the most intriguing and potential for future work and evolvement.

As we saw, "regular" models reached 56% accuracy in attempts to predict the pathology severity of a given mammogram image.

But after each mammography there is an initial assessment, BIRADS, which later if needed may be approved or disproved by a biopsy or other more deep examinations, yet we believe it is safe to assume that the assessment is relatively accurate, or at least points in the right direction usually.

So an idea arose to attempt to combine the images, with their BIRADS initial assessment, and feed the combined information to a model, so it can use the BIRADS to help itself distinguish between different pathology severity categories, and perhaps enhance correct feature extraction.

Since the BIRADS assessment is a letter: 'A': 0, 'N': 1, 'B': 2, 'P': 3, 'S': 4, 'M': 5, 'K': 6, while downloading images according to their path_severity, we also checked what BIRADS value correspond to the image's patient id, exam id and breast side, and added the image's BIRADS assessment letter as the first char in the image's name (yes the images were organized in folder following their pathology severity). This way we managed to have the best of two worlds.

As for creating a model that can extract this information and use both the image and the birads assessment we decided to go for a ViT based model, using not pretrained 'vit$_{b}ase_{p}atch16_{2}24'(avisualtransformeradaptedtoi$

ViTs are usually good for global image understanding, yet here we aim at attention to details, so why ViT?

Because we connected the use of BIRADS and the images in a late fusion flow:

1. Images enter the ViT, it analyzes them, outputting a meaningful embedding with enhanced features.
2. BIRADS to embedding.
3. Concatenate the ViT output and the BIRADS tokens.
4. Fully connected layer, the combined vector decides on class.
5. Output classes layer.

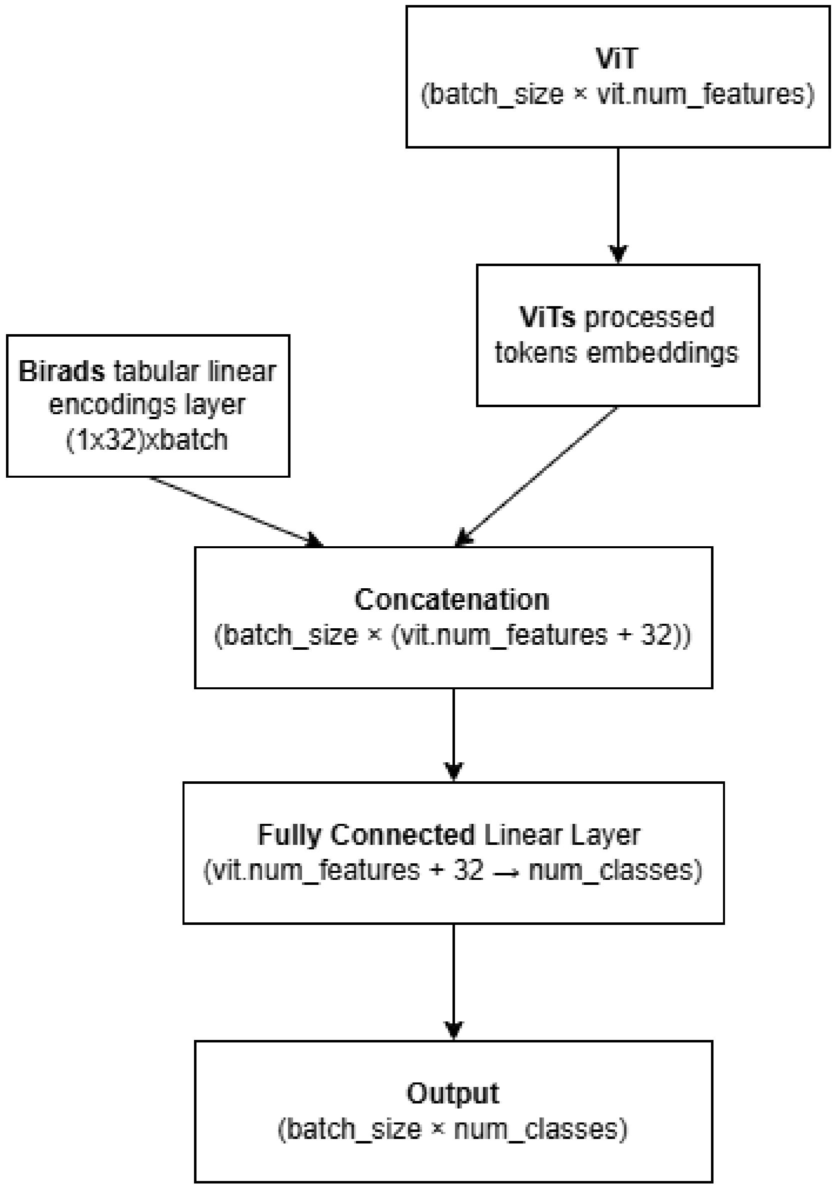

ViT
(batch_size × vit.num_features)
ViTs processed tokens embeddings
Birads tabular linear encodings layer (1x32)xbatch
Concatenation
(batch_size × (vit.num_features + 32))
Fully Connected Linear Layer
(vit.num_features + 32 → num_classes)
Output
(batch_size × num_classes)

Figure 10: A flow diagram imitating the structure of the ViT model with BIRADS token embeddings

Figure 10 imitates the algorithm's flow.
Table 8 describes the model's performance.
**Unfortunately, the three first rows in table 8 were computed on partial data which was discovered too late.
Yet still it is an impressive improvement and performance on 40 epochs, outperforming the results from 4th experiment - 56%, here 85% accuracy.

So the idea of combining BIRADS information with the images seems to be paying off!

We wanted to see whether the success was random or was the model really "looking" at a logical part of the image that can help make the correct prediction, so we computed a saliency map ("how much each pixel affects the class score") for 2 images out of every pathology severity category, and put in on the image to create a kind of a heat map similar to Grad-Cam. (We didn't want to use gradcam because in needs the gradients from the ViT attention layer, but in our case the Birads wasn't used at that level, only later so it would not have reflected the full picture of both the images' and the BIRADS' effect on the decision). See figure 11.

Epochs	ViT_Base_Patch16_224 (tabular fusion)
10 epochs	29.36* 0.6195
20 epochs	47.71*
30 epochs	48.62*
40 epochs	**85.09**** **0.9670**

Table 8: Performance of ViT_Base_Patch16_224 with on `path_severity field with concatenated BIRADS information`.

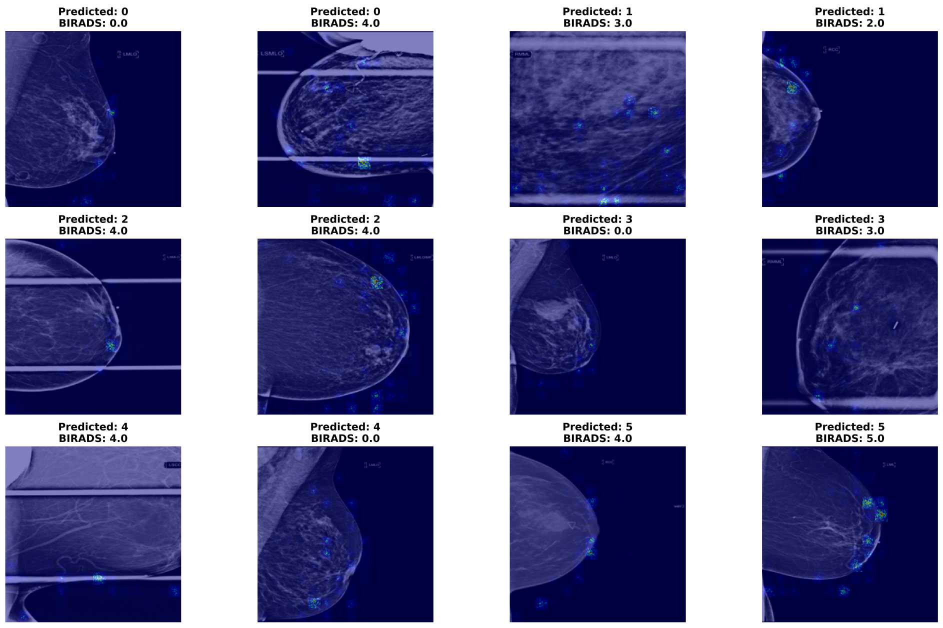


Figure 11: A saliency map of the ViT BIRADS' model's areas of interest. There are two images out of each pathology severity class, by order. Predicted class is the correct class in all images.

We see from figure 11 that the model did not focus on the background, it "looked" at important parts of the mammograms, where potentially relevant spores can be.

Figure 12 describes the confusion matrix of the ViT BIRADS model on a semi-test set containing 100 images of each pathology severity level.

Note for future work:

Here we used late fusion (concatenation of ViT images and BIRADS).

It is interesting to look what will happen in a case of early or mid fusion - before going through the ViT encoder or in the middle, having both processes.

Or after the late fusion having the concatenated information go though another network, MLP or another ViT, or some neural network.

This will also allow you to use GradCam to see the actual gradients which is a little safer reflector.

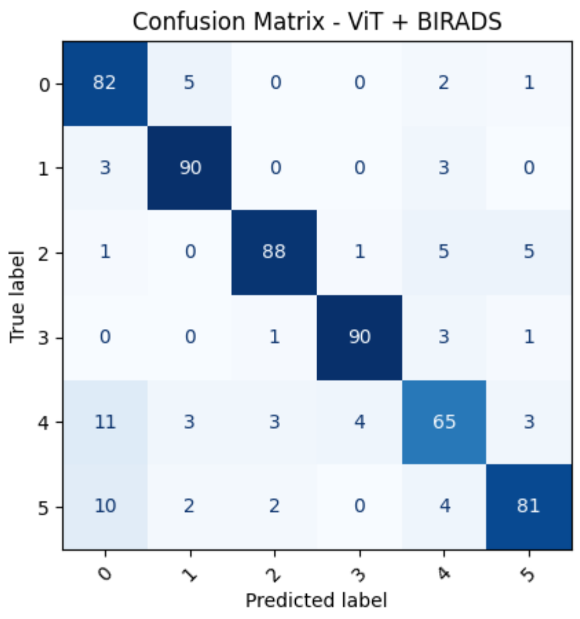


Figure 12: A confusion map of the ViT BIRADS’ model’s performance on 100 images of each pathology severity level.

6 Future work

There are couple of ideas to enhance the program’s usability and functionality in the future:

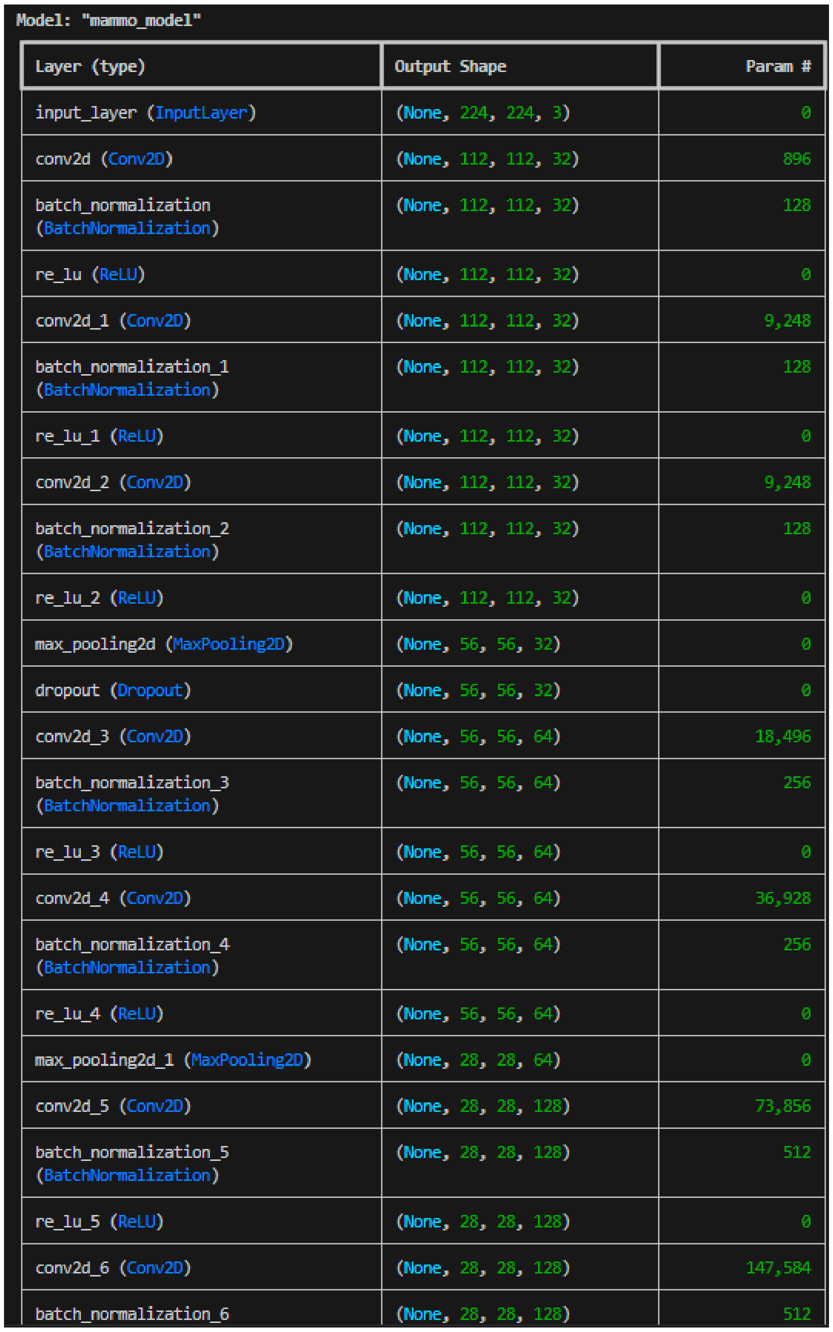
- Expanded Features: Future iterations of the viewer could benefit from additional features, such as advanced data visualization tools, enhanced filtering options, and the ability to handle more extensive datasets.
  These improvements would make the application more versatile and useful for a broader range of research applications.
- Improving the download and cache mechanism - this can improve the runtime and the storage use of the viewer, making it easier to use in personal computers without needing large storage.
- Experimenting with early or mid-fusion in the ViT model as mentioned earlier.

7 References

- EMBED Analyzer : `https://github.com/AyeletKat/EMBED-analyzer.git`
- EMBED Backend : `https://github.com/AyeletKat/EMBED-server.git`
- EMBED Frontend : `https://github.com/Oriya-Sigawy/EMBED-electron.git`
- Models used in experiments : `https://github.com/AyeletKat/EMBED-DL-Models.git`
- Breast cancer cases and deaths are projected to rise globally : `https://www.iarc.who.int/wp-content/uploads/2025/02/pr361_E.pdf`
- Understanding breast cancer as a global health concern - PMC : `https://pmc.ncbi.nlm.nih.gov/articles/PMC8822551/`
- Breast cancer : `https://www.who.int/news-room/fact-sheets/detail/breast-cancer`
- Breast cancer: Global patterns of incidence, mortality, and trends : `https://ascopubs.org/doi/10.1200/JCO.2023.41.16_suppl.10528`
- Breast Cancer Statistics And Resources : `https://www.bcrf.org/breast-cancer-statistics-and-resources/`
- Survival Rates for Breast Cancer : `https://www.cancer.org/cancer/types/breast-cancer/understanding-a-breast-cancer-diagnosis/breast-cancer-survival-rates.html`

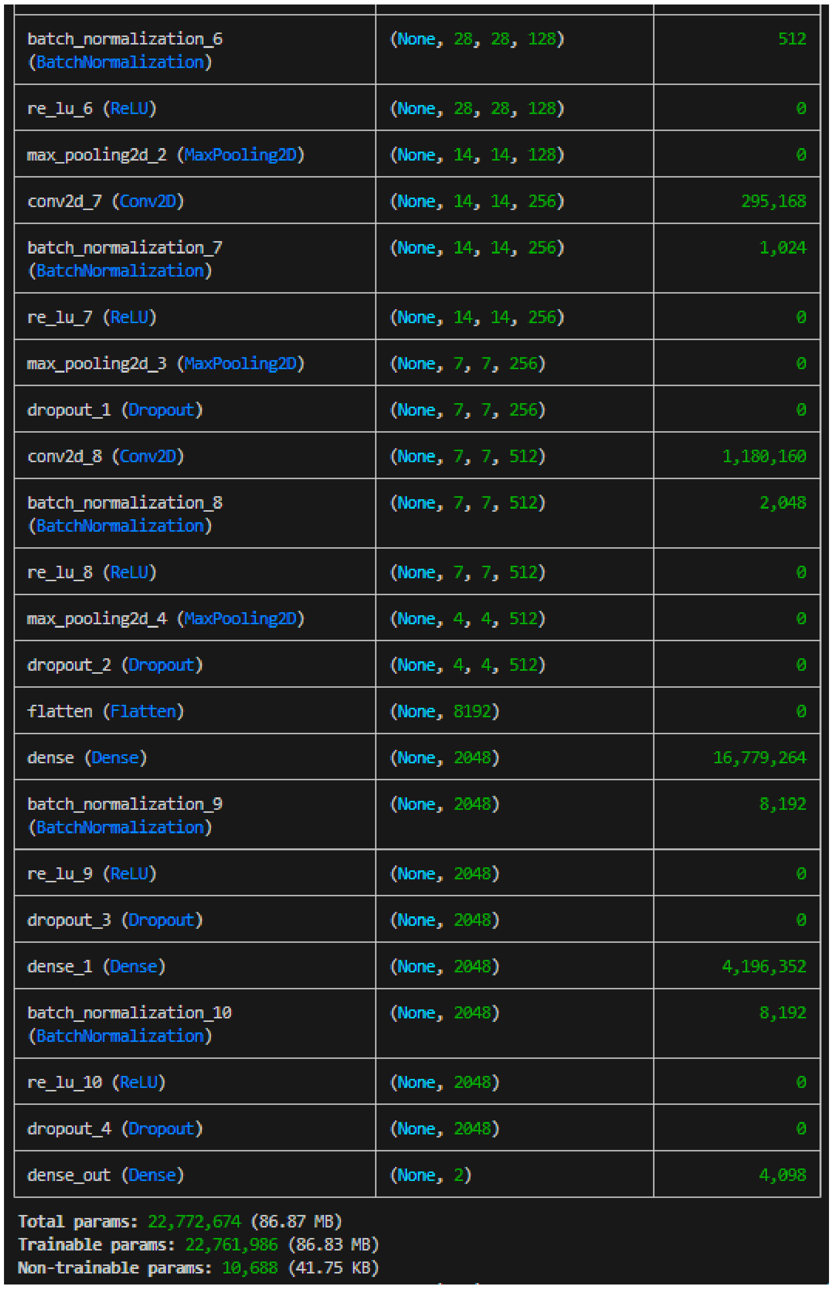
- Cancer Statistics : https://seer.cancer.gov/statistics/
- The EMBED - A racially diverse granular dataset of 3 to 4 million screening.pdf : https://www.researchgate.net/publication/366870100_The_EMory_BrEast_Imaging_Dataset_EMBED_A_Racially_Diverse_Granular_Dataset_of_34M_Screening_and_Diagnostic_Mammographic_Images
- Emory-HITI/EMBED_Open_Data for the Emory : https://github.com/Emory-HITI/EMBED_Open_Data.git
- Eric A. Scuccimarra's mammography-models Github directory, one of which models' was adapted in Section 5.3 : https://github.com/escuccim/mammography-models.git

8 Attachments



```
Model: "mammo_model"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                           ┃ Output Shape                 ┃         Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩
│ input_layer (InputLayer)               │ (None, 224, 224, 3)          │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d (Conv2D)                        │ (None, 112, 112, 32)         │             896 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization                    │ (None, 112, 112, 32)         │             128 │
│ (BatchNormalization)                   │                              │                 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ re_lu (ReLU)                           │ (None, 112, 112, 32)         │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d_1 (Conv2D)                      │ (None, 112, 112, 32)         │           9,248 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization_1                  │ (None, 112, 112, 32)         │             128 │
│ (BatchNormalization)                   │                              │                 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ re_lu_1 (ReLU)                         │ (None, 112, 112, 32)         │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d_2 (Conv2D)                      │ (None, 112, 112, 32)         │           9,248 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization_2                  │ (None, 112, 112, 32)         │             128 │
│ (BatchNormalization)                   │                              │                 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ re_lu_2 (ReLU)                         │ (None, 112, 112, 32)         │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ max_pooling2d (MaxPooling2D)           │ (None, 56, 56, 32)           │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ dropout (Dropout)                      │ (None, 56, 56, 32)           │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d_3 (Conv2D)                      │ (None, 56, 56, 64)           │          18,496 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization_3                  │ (None, 56, 56, 64)           │             256 │
│ (BatchNormalization)                   │                              │                 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ re_lu_3 (ReLU)                         │ (None, 56, 56, 64)           │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d_4 (Conv2D)                      │ (None, 56, 56, 64)           │          36,928 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization_4                  │ (None, 56, 56, 64)           │             256 │
│ (BatchNormalization)                   │                              │                 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ re_lu_4 (ReLU)                         │ (None, 56, 56, 64)           │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ max_pooling2d_1 (MaxPooling2D)         │ (None, 28, 28, 64)           │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d_5 (Conv2D)                      │ (None, 28, 28, 128)          │          73,856 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization_5                  │ (None, 28, 28, 128)          │             512 │
│ (BatchNormalization)                   │                              │                 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ re_lu_5 (ReLU)                         │ (None, 28, 28, 128)          │               0 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ conv2d_6 (Conv2D)                      │ (None, 28, 28, 128)          │         147,584 │
├────────────────────────────────────────┼──────────────────────────────┼─────────────────┤
│ batch_normalization_6                  │ (None, 28, 28, 128)          │             512 │
```

Figure 13: Reconstruction of model discussed in Section 5.3. Part I



```
│ batch_normalization_6          │ (None, 28, 28, 128)       │            512 │
│ (BatchNormalization)           │                           │                │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ re_lu_6 (ReLU)                 │ (None, 28, 28, 128)       │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ max_pooling2d_2 (MaxPooling2D) │ (None, 14, 14, 128)       │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ conv2d_7 (Conv2D)              │ (None, 14, 14, 256)       │        295,168 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ batch_normalization_7          │ (None, 14, 14, 256)       │          1,024 │
│ (BatchNormalization)           │                           │                │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ re_lu_7 (ReLU)                 │ (None, 14, 14, 256)       │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ max_pooling2d_3 (MaxPooling2D) │ (None, 7, 7, 256)         │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dropout_1 (Dropout)            │ (None, 7, 7, 256)         │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ conv2d_8 (Conv2D)              │ (None, 7, 7, 512)         │      1,180,160 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ batch_normalization_8          │ (None, 7, 7, 512)         │          2,048 │
│ (BatchNormalization)           │                           │                │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ re_lu_8 (ReLU)                 │ (None, 7, 7, 512)         │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ max_pooling2d_4 (MaxPooling2D) │ (None, 4, 4, 512)         │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dropout_2 (Dropout)            │ (None, 4, 4, 512)         │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ flatten (Flatten)              │ (None, 8192)              │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dense (Dense)                  │ (None, 2048)              │     16,779,264 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ batch_normalization_9          │ (None, 2048)              │          8,192 │
│ (BatchNormalization)           │                           │                │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ re_lu_9 (ReLU)                 │ (None, 2048)              │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dropout_3 (Dropout)            │ (None, 2048)              │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dense_1 (Dense)                │ (None, 2048)              │      4,196,352 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ batch_normalization_10         │ (None, 2048)              │          8,192 │
│ (BatchNormalization)           │                           │                │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ re_lu_10 (ReLU)                │ (None, 2048)              │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dropout_4 (Dropout)            │ (None, 2048)              │              0 │
├────────────────────────────────┼───────────────────────────┼────────────────┤
│ dense_out (Dense)              │ (None, 2)                 │          4,098 │
└────────────────────────────────┴───────────────────────────┴────────────────┘
 Total params: 22,772,674 (86.87 MB)
 Trainable params: 22,761,986 (86.83 MB)
 Non-trainable params: 10,688 (41.75 KB)
```

Figure 14: Reconstruction of model discussed in Section 5.3. Part II