Ατομική Διπλωματική Εργασία

BREST CANCER LOCATION DETECTION WITH MACHINE LEARNING USING THERMOGRAPH IMAGES

Damianos Kallis

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μάιος 2025

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΥΠΡΟΥ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Brest cancer location detection with machine learning using Thermograph images

Damianos Kallis

Επιβλέπων Καθηγητής
Christodoulou Christos

Η Ατομική Διπλωματική Εργασία υποβλήθηκε προς μερική εκπλήρωση των απαιτήσεων απόκτησης του πτυχίου Πληροφορικής του Τμήματος Πληροφορικής του Πανεπιστημίου Κύπρου

Μάιος 2025

Acknowledgements

First, the main idea came from MammoCheck Ltd and is the one that gave me this opportunity for this experience. Also, I would like to thank Prof Christos Christodoulou for getting me in touch with the company and helping me with my progress, but he also taught me the basics of Machine Learning, and Computational Intelligence. But the most help I got was from one of the owners of the company Marios Pafitis who helped me to understand each phase of the project and gave me ideas how to continue my research each time I had a problem and provided me with useful datasets for my project. Furthermore, I would like to thank Valentinos Pariza for helping me to set up Dinov2 and tutor me how it works, by giving me a template and explaining each function and Marios Constantinou from the HPC Administrators team that gave me access to the HPC. Lastly, I would like to thank my family and friends that support and help me get through this project .

Abstract

Many people today have been diagnosed with breast cancer and in most of the cases the late detection causes the cancer to be incurable. Today there are various ways to diagnose breast cancer like mammography and ultrasound breast exams however they are not taken so frequently from the patients because they are expensive, dangerous or painful. As a result, recent studies presented thermography as a complementary tool to primary diagnostic methods, for frequent monitoring, which is easier to take, pain-less and radiation-free which can be conducted at home in few minutes using an affordable thermal camera attached to a smartphone. However, many studies showed that with thermography and Artificial Intelligent programs they can detect if a patient has breast cancer, but they cannot detect the location of the cancer. So, the goal of my study is to create an algorithm using machine learning to detect the location of the cancer in breast thermal images and compare different hyperparameters and analyse the results to see how the model performs for this task.

Περιεχόμενα

Chapter 1

Introduction

1.1 Problem with common examinations

Breast cancer accounts for one in three female cancers in the United States. The American Cancer Society estimates that, of the approximately 168,260,000 women in the US, 316,950 will receive a new diagnosis of invasive breast cancer annually, and 42,170 will lose their lives to the disease [1]. Mammograms, ultrasounds, breast magnetic resonance imaging (MRI), and other primary breast cancer screening procedures are typically performed on a periodic basis on women. The most crucial and gold standard test for breast cancer is mammography, which can identify tumours up to two years before they are palpable. However, it is advised that women have routine screenings every two to three years, beginning at age 45 [2], due to the ionizing radiation that mammography exposes the patient to. Another screening method is ultrasound, which calls for a specialist to operate the equipment and interpret the findings. [3]

1.2 Problem with Thermography

On the other hand, thermography is an FDA-approved adjunct to primary breast screening methods for breast cancer examinations [4]. The primary advantage of thermography is that it can be carried out using reasonably priced thermal camera equipment, and as the camera does not produce any radiation, it poses no risk to the patient. Unfortunately, a lot of research added to the market models for cancer detection showing if a patient is diagnosed with breast cancer or not but there is not a model showing the exact location of the cancer. [3]

1.3 Thesis’ goal

As I previously mentioned it is important for fast treatment of the breast cancer to be early detected but it is also important for the right treatment to have early localization of the tumour area. In this thesis I will try to address this issue, by trying with different models, different data and data preprocessing to create an unsupervised algorithm that combines multiple techniques to be able to achieve the best accuracy with the detection of the cancer area on ill women but also see how the program reacts with healthy women and analyse my results using specific metrics.

Chapter 2

Background

2.1 Machine learning Clustering algorithms

Cluster algorithms are models that separate different inputs into groups based on the similarity of their features that we have in the dataset. Those models can be supervised or unsupervised

2.1.1 Supervised Cluster algorithms

In supervised algorithms we have in the dataset the desired cluster output for each input, and it works like 'teacher' providing the net with the output required for each input. This algorithm learns by adjusting the weights to minimize the difference between the desired and actual outputs for each input pattern.

2.1.1.1 Random Forest Classifier

The basic idea in decision tree classifier is the separation of the inputs at each non-terminal node and the choice of the most important data features for the separation of the inputs.

Random Forest is a popular and easy machine learning algorithm used for several types of classification tasks. A Random Forest is a combination of tree-structured classifiers. Every tree of the forest gives an output , assigning each input to the closest class label. As we see in the [Figure 1] there are 4 decision trees, and each tree takes as input from the dataset x and based on different features each tree gives an output, and the most common output is the "winner". It is a fast method, robust to noise and it is a successful ensemble which can identify non-linear patterns in the data. It can easily handle both numerical and categorical data. One of the major advantages of Random Forest is that it does not suffer from over-fitting, even if more trees are appended to the forest. [5]

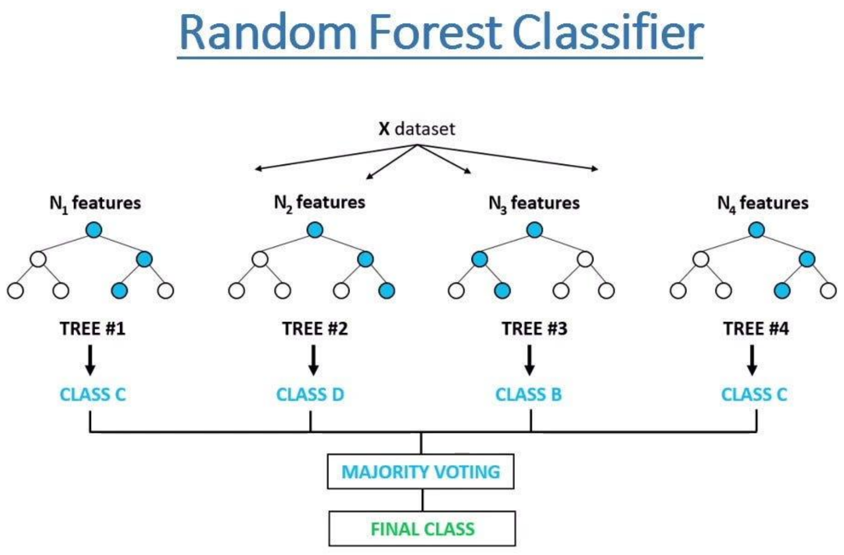


Figure 1:Random Forest Classifier diagram with 4 trees

2.1.2 Unsupervised Cluster algorithms

Unlike supervised in unsupervised algorithms we do not have the desired output of each input [6]. This network can discover statistical regularities in its input space and automatically develops different modes of behavior to represent different classes of inputs (in practical applications some `labelling' is required after training, since it is not known at the outset which mode of behavior will be associated with a given input class).[5]

2.1.2.1 K-means algorithm

K-means is a simple unsupervised cluster algorithm and it is very common as it solves most of the clustering problems. The main idea is to define k centroids, one for each cluster. We place these centroids most of the time random but different location causes different results. So, if it's not random we try to place them as much as possible far away from each other. Next step is to take each point from the inputs and associate it with the closest centroid and when there are no inputs left the first epoch is finished. Then we re-calculate the k centroids at the centre of each cluster, and we repeat the association with the same inputs as previous and the new centroids and we repeat this process until the k centroids stop changing locations. That means that the clusters have the minimum square error [7]. And we can see an example of 2D point that are pass through K-Means with 3 as K and it cluster the point into 3 cluster based on the position [Figure 2].

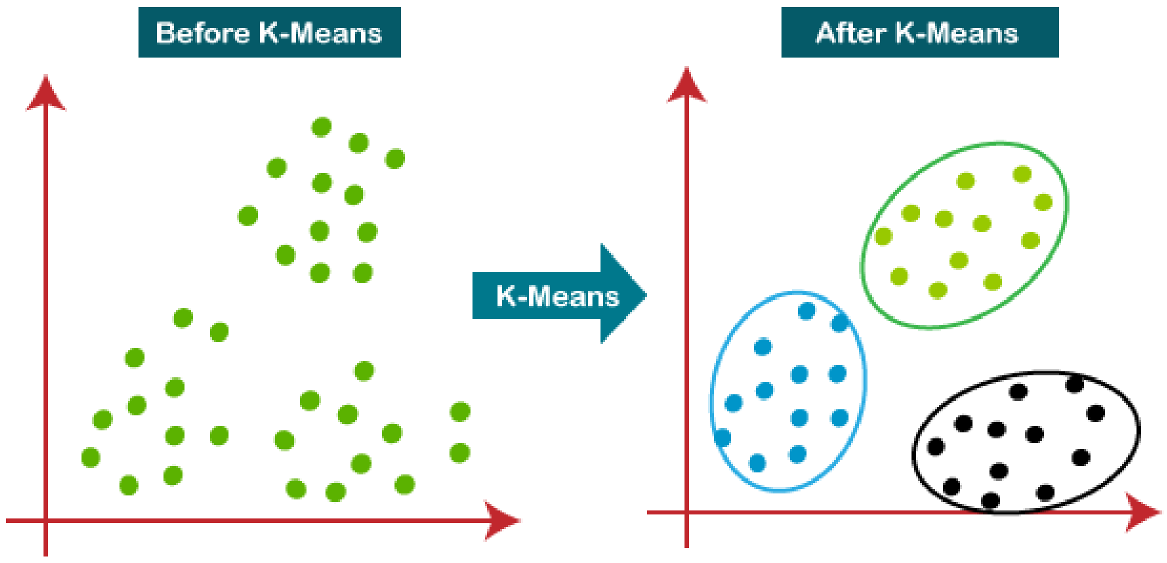


Figure 2: 2D Point diagram before and after K-Means classification

2.2 Vision Transformer

The Vision Transformer (ViT) introduces a new era of computer vision where images are treated as sequences of patches, just like words in Natural Language Processing (NLP). By leveraging Transformers instead of Convolutional Neular Networks (CNNs), ViT achieves state-of-the-art performance on image recognition tasks with large datasets. While it requires large datasets for pre-training, its simplicity, scalability, and effectiveness make it a compelling alternative to CNNs. This paradigm shift from convolutional processing to attention-based processing could inspire new models for object detection, segmentation, and other vision tasks. Registers are extra learnable tokens added to the Vision Transformer input sequence that provide dedicated space for internal computations, preventing the model from corrupting patch features and improving attention smoothness and downstream task performance. Without registers, vision transformers like DINOv2 models tended to repurpose background patch tokens (with little information) for internal computations, leading to degraded local features and problems like messy attention maps [Figure 3]. [8]

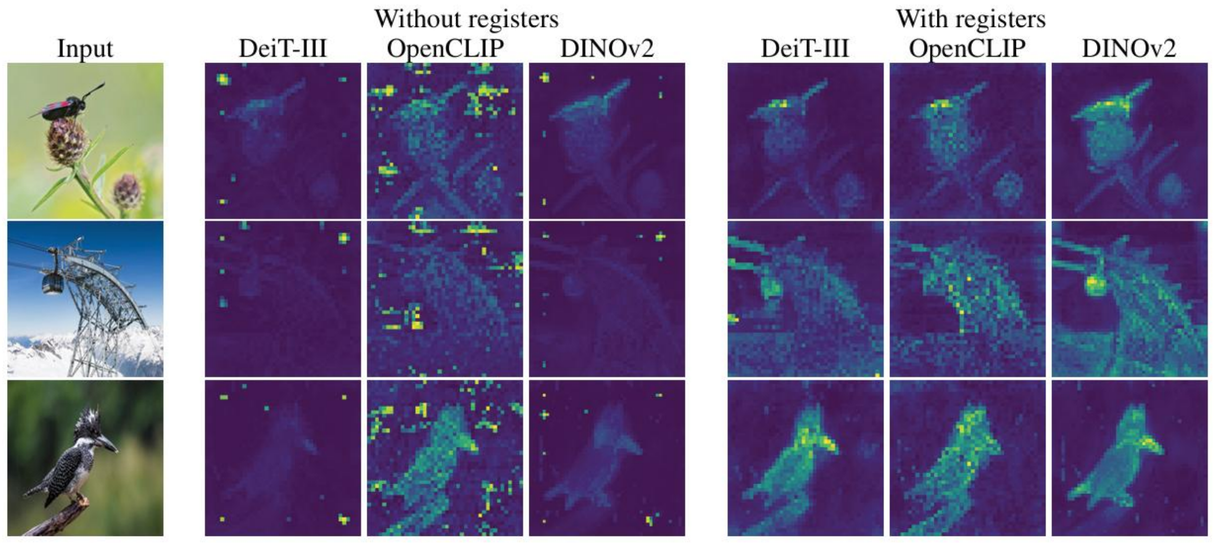


Figure 3: Register tokens enable interpretable attention maps in all vision transformers, like the original DINO method (Caron et al., 2021). Attention maps are calculated in high resolution for better visualisation. [8]

2.3 Dinov2

DINOv2 is a Self-supervised Vision Transformer Model created by Meta AI and produces universal features suitable for image-level visual tasks (image classification, instance retrieval, video understanding) as well as pixel-level visual tasks (depth estimation, semantic segmentation). The pipeline Dinov2 follows is easy as it takes only 3 steps. First it chops the uncurated data and the curated data into patches called embeddings then it removes form the uncurated data very similar images to avoid redundancy and reduce noise and last the remaining uncurated images are matched to curated images based on embedding similarity and it retrieves them [Figure 4]. [9]

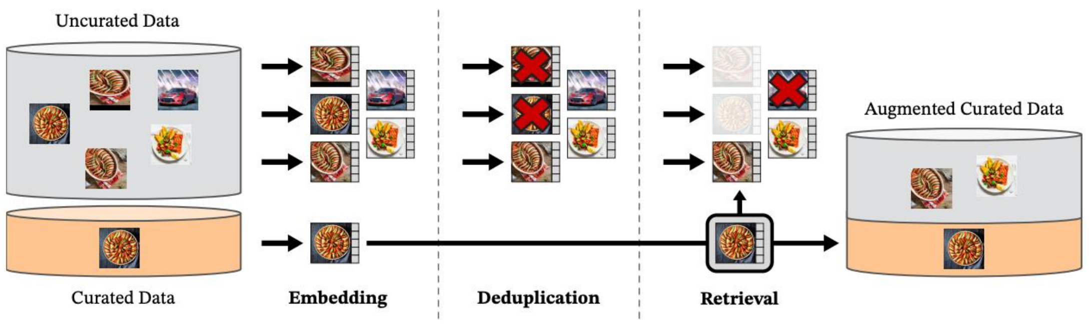


Figure 4: Overview of our data processing pipeline. Images from curated and uncurated data sources are first mapped to embeddings. Uncurated images are then deduplicated before being matched to curated images. The resulting combination augments the initial dataset through a self-supervised retrieval system.[9]

DINOv2 demonstrates that self-supervised models trained on curated datasets can achieve better generalization and robustness than weakly supervised models like Contrastive Language-Image Pre-training (CLIP). The robust features learned by DINOv2 support a wide range of vision tasks at image and pixel levels, making them suitable for classification, segmentation, depth estimation, and more. For example, it can classify different species of animals like flying animals or ground animals like horses, and it can even classify different objects like car and show also with different colours similar components between the objects in each group [Figure 5]. [9]

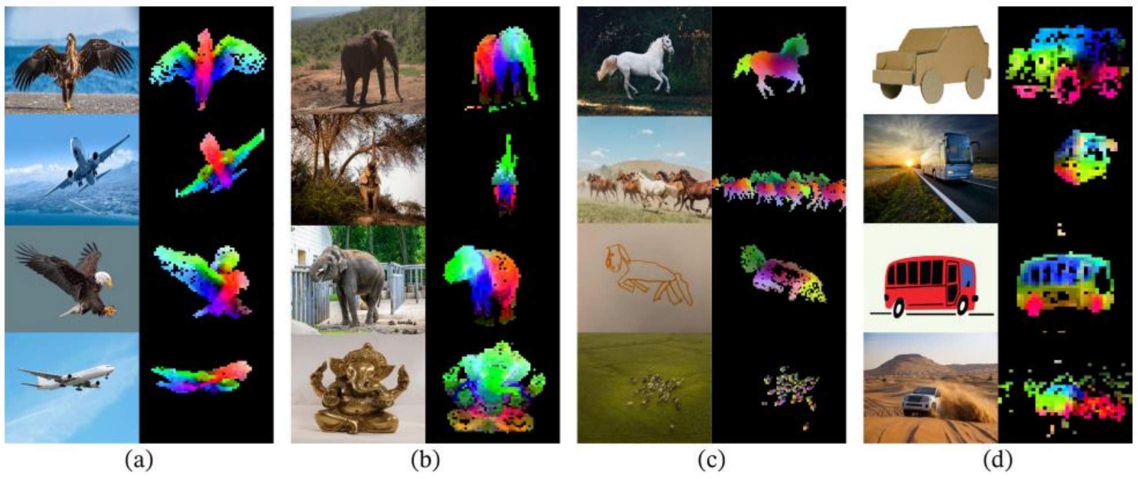


Figure 5: Visualization of the first PCA components. A PCA between the patches of the images is been computed from the same column and show their first 3 components. Each component is matched to a different colour channel. Same parts are matched between related images despite changes of pose, style or even objects.[9]

2.4 Metrics

For the task of the classification evaluation, typically we are only interested in binary ratings, that is, either the item was selected (1) or not (0) or in our case the patch has a tumour (1) or not (0). [10] This matrix contains statistics about actual and predicted classifications and lays the fundamental foundations necessary to understand accuracy measurements for a specific classifier.[11]

2.4.1 TP,FP,TN,FN

This matrix is characterized with 4 attributes that we can compute based on the desired and the actual output where we check whether the cluster that represents the cancer area has the same patches as the label we created. The attributes are the follows:

- TP: A patch in the tumour cluster belong in the tumour patches in the label file
- FP: A patch in the tumour cluster does not belong in the tumour patches in the label file
- TN: A patch that is not in the tumour cluster does not belong in the tumour patches in the label file
- FN: A patch that is not in the tumour cluster belong in the tumour patches in the label file

		Actual Values	
		Positive	Negative
Predicted Values	Positive	TP	FP
	Negative	FN	TN

Table 1: Confusion Matrix for Binary Classifier

2.4.2 Accuracy

Accuracy is the fraction of correct predictions among all predictions or how often a prediction is correct. However, the classification accuracy has limitations, especially when dealing with imbalanced datasets, where instances of one class may significantly outnumber those of other classes. Accuracy = (Number of correctly classified instances) / (Total number of instances) [12]

Accuracy = (TP + TN) / (TP + FP + TN + FN)

2.4.3 Precision

If there are more than two output classes, the classification accuracy does not provide information on the classes that are predicted accurately. In such cases, a more suitable measure is precision. Precision is the fraction of the correctly predicted positive results. To be more specific it shows the percentage of how many of the patches in the tumour cluster are actual tumour patches. [12]

Precision = TP / (TP + FP)

2.4.4 Recall

This measures the proportion of actual positives predicted correctly, or how accurately the model predicts positive cases. Recall is particularly valuable when false negatives need to be minimized. For example, in a medical diagnostic setting like in our problem, recall indicates the proportion of actual positive cases that the model correctly identified, which is important to avoid missing potentially critical diagnoses. [12]

Recall = TP / (TP + FN).

2.4.5 F1-score

The F1-score, which combines precision and recall, is important for models in which both are equally important and models for healthcare applications are perfect examples. This is a harmonic means of precision and recall that considers both false negatives and false positives. The harmonic mean is computed by dividing the number of values in a data series by the sum of the reciprocals of each value in the data series. [12]

F1-score = (2 * precision * recall) / (precision + recall)

2.4.6 Mean Average Precision

Mean Average Precision (mAP) is the most common evaluation metric. Precision is derived from Intersection over Union (IoU), which is the ratio of the area of overlap and the area of union between the ground truth and the predicted bounding box. A threshold is set to determine if the detection is correct. If the IoU is more than the threshold, it is classified as True Positive while an IoU below it is classified as False Positive. If the model fails to detect an object present in the ground truth, it is termed as False Negative. Precision measures the percentage of correct predictions while the recall measures the correct predictions with respect to the ground truth.[13]

Chapter 3

Data Handling

3.1 Brest Thermography

The idea behind breast thermography is to use an infrared (IR) camera to monitor the breasts' surface temperature and then use image post-processing to pinpoint regions of a normal temperature. In 1956, Lawson provided one of the earliest indications that breast cancer alters the breasts' typical temperature distribution [14]. In comparison to the contralateral region of the breast without a tumour, all 26 participants with breast cancer in this study had an increase of 1.3 C close to the tumour location. And later studies showed even more increase in the temperature of the area around the tumour [15].

3.1.1 “Breast Thermography” dataset

It is image thermography dataset of female thorax zone as scientific contribution in breast cancer thermography research area. Dataset was acquired inside a medical office with dimensions W:3.20m x L:4.14m x H:2.40m, in grayscale without artificial illumination at a temperature 22 - 24°C and relative humidity 45 - 50%. A FLIR A300 camera was used to capture the images at a working distance of 1m from the patient. The American Academy of Thermology (AAT) protocol was used to prepare the patient and capture. Three images were taken of each patient in the female thorax area: anterior, left oblique and right oblique positions, but I used only the anterior so that I have the same angle for all my data. The patients ranged in age from 18-81 years with different pathologies related to the female breast. The images were captured between 2021 and 2022. There is also a csv file that indicate which breasts are malign or benign or normal and has also some information of the patient like age, height e.t.c [16]. Using a plot, we can see an example of 32 anterior images from this dataset with 32 ill women where we can recognise that in grayscale the tumour area is brighter from the other areas [Figure 6] and we can also see the difference with the plot with the 32 anterior images from 32 healthy woman [Figure 7].

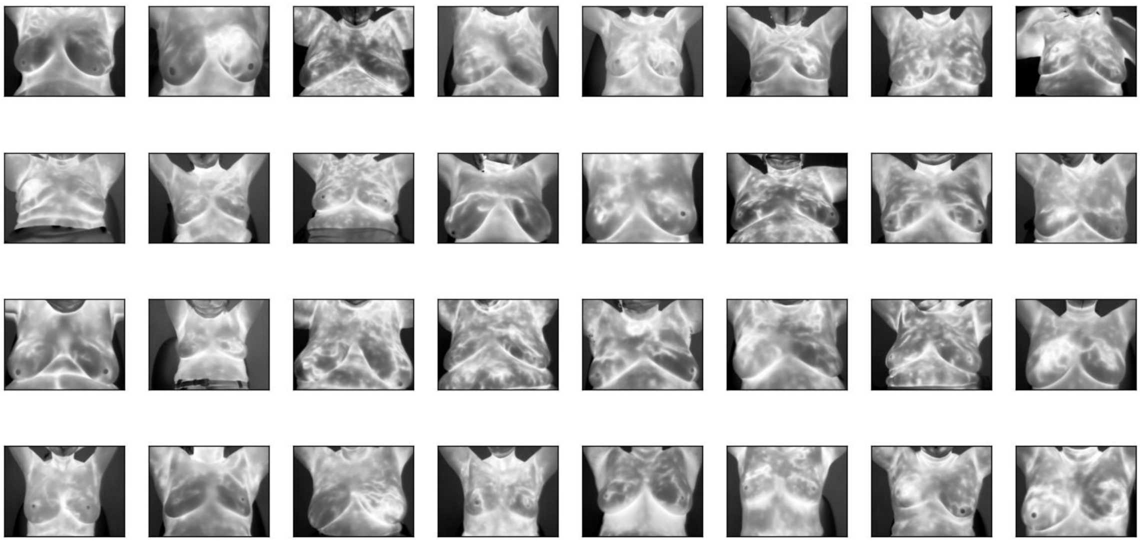


Figure 6: 32 of anterior images of ill women from "Breast Thermography" dataset

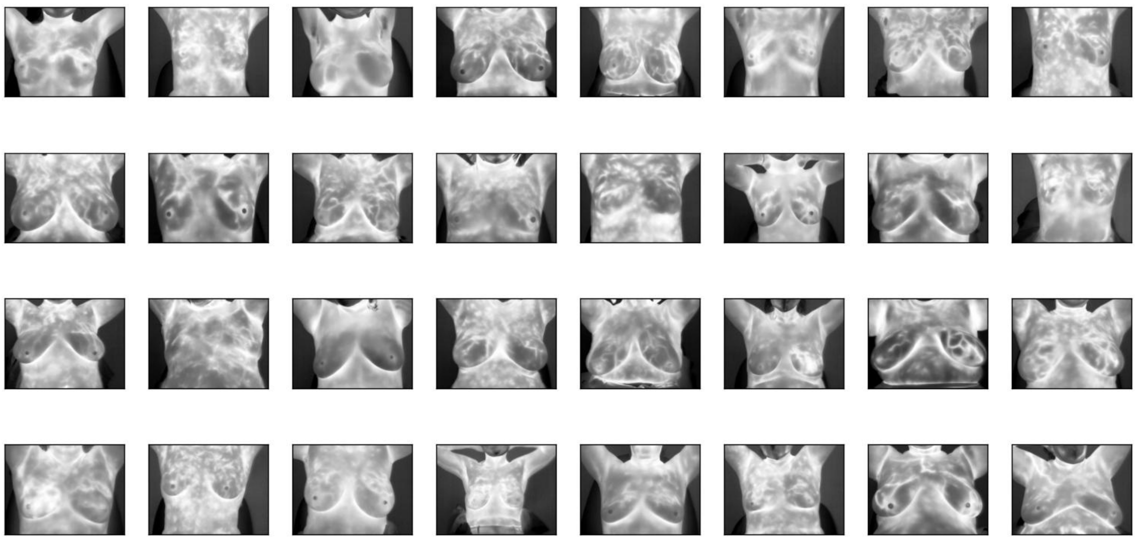


Figure 7: 32 of anterior images of healthy women from "Breast Thermography" dataset

3.1.2 “A New Database for Breast Research with Infrared Image” dataset

The dataset for the following experiments was created using the breast thermography database by Silva et al. (2014) and is the one that most researchers in breast thermography uses. The database consists of grayscale thermal images of resolution 640 × 480 pixels. The database has 173 healthy and 37 sick front-view images. The infrared (IR) images are captured by a FLIR thermal camera, model SC620, which has sensitivity of less than 0.04ºC range and capture standard −40ºC to 500ºC. [17]

3.2. Dataset usage

In my study I mostly used the “Breast Thermograph” dataset because it includes a csv file which was very helpful because, it had information for each breast if it was normal, malign pathology or benign pathology and that helped me to understand my data and have an idea where the tumour is located. I also use that information to find a more specific area of the tumours and label it. I only use “A New Database for Breast Research with Infrared Image” dataset to try the unsupervised algorithm with more data because the “Breast Thermograph” dataset has only 35 images will ill woman. Furthermore, I use it in the supervised algorithm because the labelled data I used were from the “Breast Thermograph” dataset and I wanted to use the model on different data so that the patches that I want to classify have similar features with the labelled and not the same.

3.3 Data Preprocessing

I preprocess the images using dinov2 in Google Collab where I divided the images to patches and extract their features. I used different patches combinations (15X15, 25X25 and 40X40 patches) [Table 2] with 384 features for each patch creating a variety of feature dimensions like (35,1600,384), where I had 35 images with 40X40 patches with 384 features for each patch. I also try different feature processing ideas like reduction using the Principal Component Analysis (PCA) which is a dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional space while preserving as much variance (information) as possible and I use it to reduce the 384 features to 128 and this gives the results better generalization but it losses some information as we can see with PCA reduction the breast images are less complicated [Figure 8] and the same images without PCA reduction have more information on them [Figure 9].

Patches Dimensions	Number of Patches
15X15	225
25X25	625
40X40	1600

Table 2: Dinov2 patch Maps

Algorithm 1: DINOv2 Feature Extraction Pipeline

```
   Data: Images
   Result: Patch-level features from DINOv2
1  Initialize: Number of images to process (N_IMGS).
2  Initialize: Number of rows and columns for image plotting (if needed).
3  Initialize: How many patches to extract along height and width.
4  Initialize: Choose the model size (e.g., small, base, large, giant).
5  Initialize: Set flags for using special DINOv2 versions (e.g., registers, extended).
6  Load the appropriate DINOv2 model (with or without registers).
7  Load model weights if using non-extended version.
8  Resize, center-crop, normalize, and convert images to tensor format.
9  Use a Gaussian blur and scale images to match patch dimensions.
10 Initialize: An empty tensor to store preprocessed images.
11 For each image:
12     Open it and convert to RGB.
13     Apply transformations and add to image tensor.
14 end
15 Move model and tensors to GPU (if available).
16 Disable gradient calculation for efficiency.
17 Forward images through the model.
18 Extract patch-level features from model output.
```

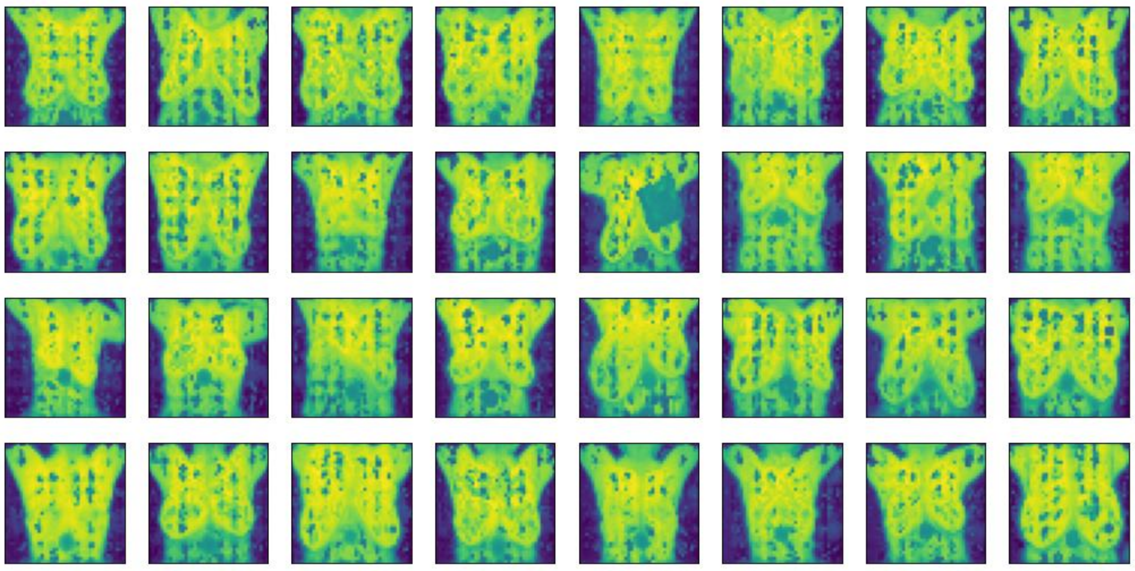


Figure 8: We can see 32 of anterior images of ill woman from "A New Database for Breast Research with Infrared Image " dataset after the algorithm using 40x40 patches and PCA reduction with to 128 features and we see how it focuses on the body and the important areas

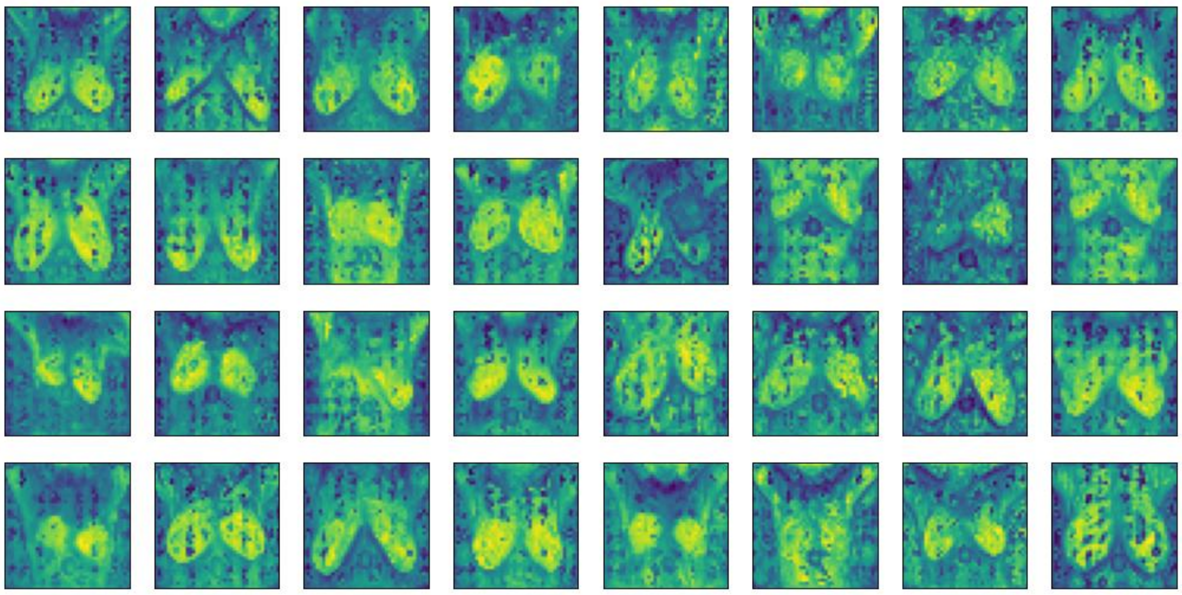


Figure 9: We can see with the same dataset as previous and 40X40 patches but without PCA reduction we have 384 features, and we can see how different the results are.

3.4 Ground Truths

As I will later mention because of the lack of information and labels of the cancer areas I had to manually label where is the tumour in the data from the ill women in the "Breast Thermography" dataset to have a desired output for evaluation. Unfortunately, I did not have an expert diagnose on the images so I found and labelled where I believe that the tumour area is located which is the red square you can see in the plot [Figure 10].

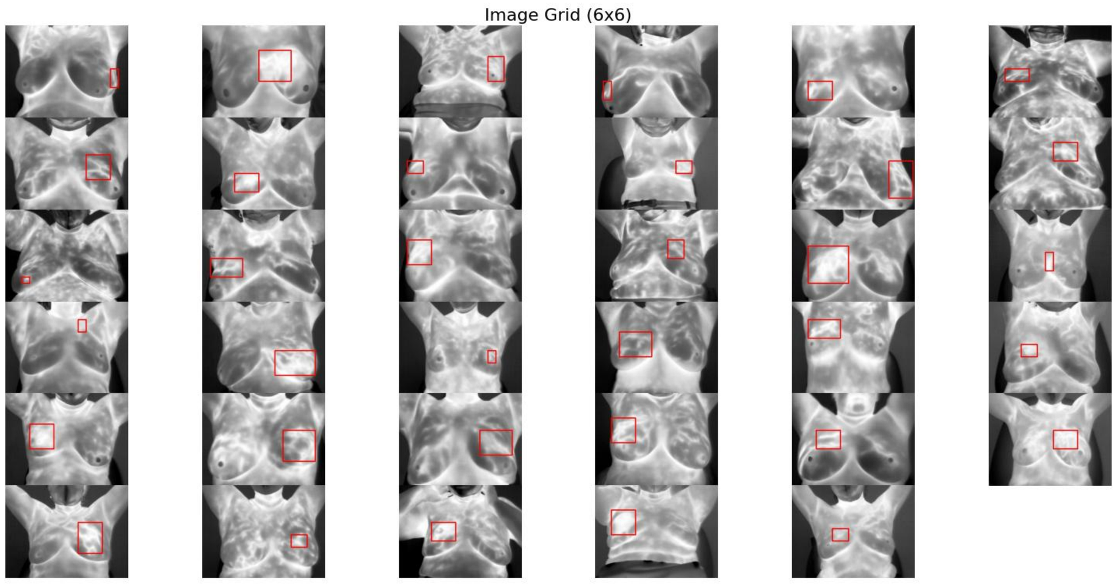


Figure 10:Plot with Ground Truths with the 36 images for the "Breast Thermography" dataset

Chapter 4

Implementation

4.1 Dinov2 features processing

First, I had to make the appropriate changes to the features regarding their shape and their normalization, and the dimension of the features based on if I want to use PCA reduction, to be able to run the clustering algorithm. So first I had to load the feature file and extract the number of images ,the number of patches and the feature dimension and in most of the experiments the feature file has the shape (number of images, number of patches , number of features per patch). Then I had to flatten the features to have shape: (number of images * number of patches, number of features per patch) and cluster the patches instead of entire images and then if I wanted to use PCA reduction I had to set the components to 128 and I use the function fit_tranform to apply it on the features.

4.2 K-means implementation

Then for the clustering I used K-means using as input the normalised 2D features as I described. I used multiple numbers of cluster, and random_state=42 to ensure reproducible results and I used 10 as n_init that initialise how many times it will run the algorithm with different centroids to get the best result. The model then gives for output a 1D array of length image_num * patches_num where each value is the cluster number of the specific patch. But having all the patches' cluster number in a 1D was not helpful so I had to reshape the array to (N_IMGS, NUM_PATCHES) to be able to set some coordinates to the patches to be easier to work with. After the training is over, we can see the visualization of the clusters and analyze the results to try different ideas and parameters [Figure 11].

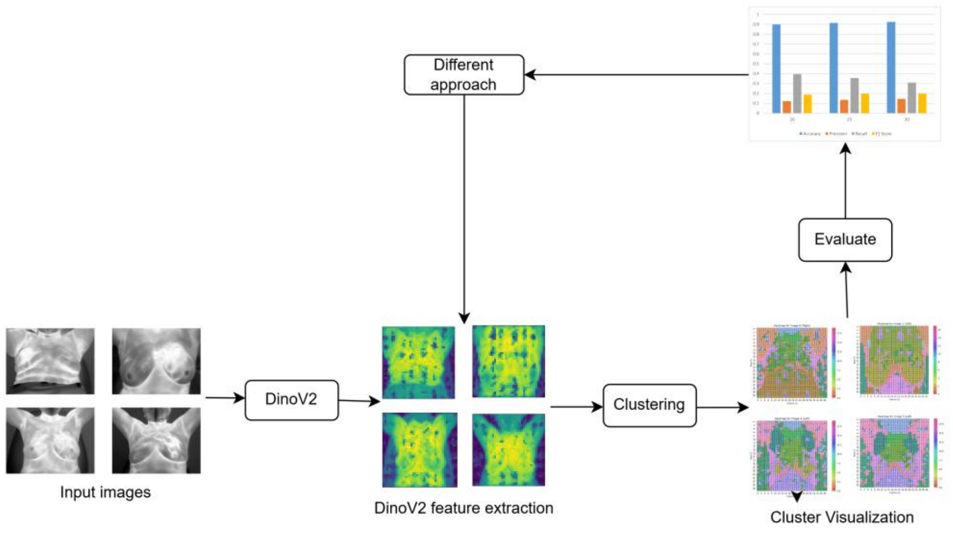


Figure 11:Algorith pipeline

4.3 Labelling for supervised classification

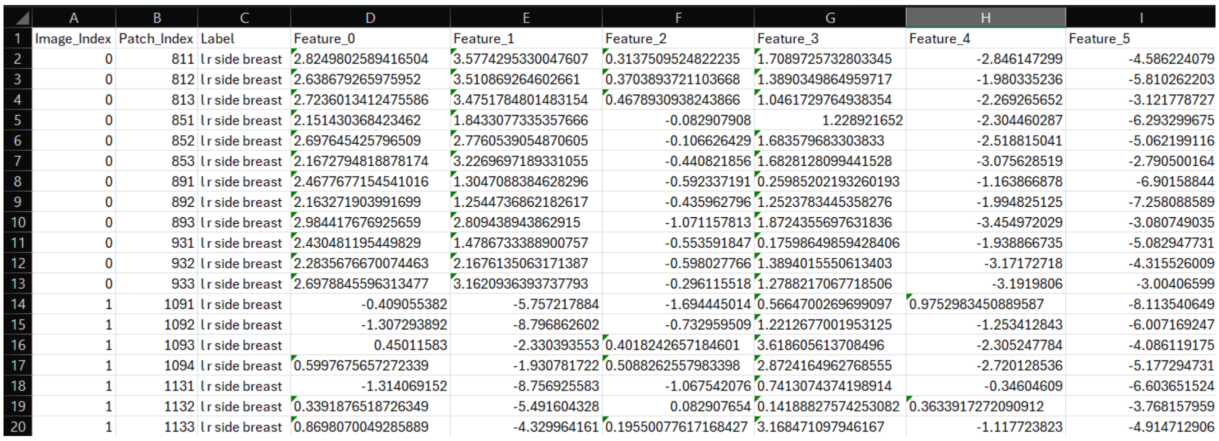
Because of the limited data information I wasn't able to find the desired cancer location and labels from an expert and I had to create my own manual label so I crated a program for that where I load the images, and display them one by one and select the area that I believe it was the cancer area and label it using the labels :l l side breast, l r side breast, r r side breast, r l side breast, r nipple, l nipple, l upper breast, r upper breast where r is right and l is left and for example l r is the left breast right side [**Table 3**]. Then the image index , the index of the patches that were in the area I selected, and the label and their 384 features were saved in an csv file that I will use later [Figure 12].

Label Name	Description
l l side breast	Left breast at the left side area
l r side breast	Left breast at the right-side area
r r side breast	Right breast at the right-side area
r l side breast	Right breast at the left side area
r nipple	Right breast at the nipple area
l nipple	Left breast at the nipple area
l upper breast	Left breast at the upper area
r upper breast	Right breast ate the upper area

Table 3: Supervised labels for the area of the tumour

Algorithm 2: Labelling for supervised classification

```
     Data: A Folder With images, Patch-level features from DINOv2 for the images
     Result: CSV file with the patches number and their features and tumour area label
1    Define: list of feature files (e.g., "dinov2_features_sick2_15.pt")
2    Load all .pt files using torch.load()
3    Concatenate them into a single tensor called 'features'
4    Derive NUM_PATCHES per image
5    Derive FEATURE_DIM (number of features per patch)
6    Derive N_IMGS from total number of patches and NUM_PATCHES
7    Set: image_folder path
8    Read and sort all image filenames from the folder with valid extensions (.png, .jpg,
     .jpeg)
9    PATCH_H ← sqrt(NUM_PATCHES)
10   PATCH_W ← sqrt(NUM_PATCHES)
11   Define: save_path for CSV (e.g., "Labelled_patch_features3.csv")
12   If CSV doesn't exist:
13       Create it with column headers: ["Image_Index", "Patch_Index", "Label", Feature_0,
    ... Feature_N]
end
on_click(event):
    Store starting coordinates on first click
    Store ending coordinates and calculate region on second click
While True:
    Print available images with their indices
    Ask user to input an image index (or -1 to exit)
    If user enters -1:
        Exit program
    end
    Validate the index:
        If invalid, show an error and repeat
    end
    Load the selected image
    Compute patch size dynamically from actual image dimensions
    Show image using matplotlib
    Let user select a rectangular region (2 mouse clicks)
    Ensure width & height are positive (in case of reverse dragging)
    Find all small patches within selected rectangle:
    For each NUM_PATCHES:
        Compute patch_x and patch_y based on row and column in grid
        Check if the patch falls inside the selected rectangle
        If yes, add patch_idx to selected_patches
    end
    Show image again with red rectangle (user selection) and blue boxes (selected
    patches)
    Ask user for a label (e.g., “l under breast”, “l upper breast”, etc.)
    For each selected patch:
        Extract the corresponding feature vector from the features tensor
        Append [Image_Index, Patch_Index, Label, Feature Vector] to a list
    end
    Convert list to DataFrame and append it to the CSV file
end
```



	A	B	C	D	E	F	G	H	I
1	Image_Index	Patch_Index	Label	Feature_0	Feature_1	Feature_2	Feature_3	Feature_4	Feature_5
2	0	811	l r side breast	2.8249802589416504	3.5774295330047607	0.3137509524822235	1.7089725732803345	-2.846147299	-4.586224079
3	0	812	l r side breast	2.638679265975952	3.510869264602661	0.3703893721103668	1.3890349864959717	-1.980335236	-5.810262203
4	0	813	l r side breast	2.7236013412475586	3.4751784801483154	0.4678930938243866	1.0461729764938354	-2.269265652	-3.121778727
5	0	851	l r side breast	2.151430368423462	1.8433077335357666	-0.082907908	1.228921652	-2.304460287	-6.293299675
6	0	852	l r side breast	2.697645425796509	2.7760539054870605	-0.106626429	1.683579683303833	-2.518815041	-5.062199116
7	0	853	l r side breast	2.1672794818878174	3.2269697189331055	-0.440821856	1.6828128099441528	-3.075628519	-2.790500164
8	0	891	l r side breast	2.4677677154541016	1.3047088384628296	-0.592337191	0.25985202193260193	-1.163866878	-6.90158844
9	0	892	l r side breast	2.163271903991699	1.2544736862182617	-0.435962796	1.2523783445358276	-1.994825125	-7.258088589
10	0	893	l r side breast	2.984417676925659	2.809438943862915	-1.071157813	1.8724355697631836	-3.454972029	-3.080749035
11	0	931	l r side breast	2.430481195449829	1.4786733388900757	-0.553591847	0.17598649859428406	-1.938866735	-5.082947731
12	0	932	l r side breast	2.2835676670074463	2.1676135063171387	-0.598027766	1.3894015550613403	-3.17172718	-4.315526009
13	0	933	l r side breast	2.6978845596313477	3.1620936393737793	-0.296115518	1.2788217067718506	-3.1919806	-3.00406599
14	1	1091	l r side breast	-0.409055382	-5.757217884	-1.694445014	0.5664700269699097	0.9752983450889587	-8.113540649
15	1	1092	l r side breast	-1.307293892	-8.796862602	-0.732959509	1.2212677001953125	-1.253412843	-6.007169247
16	1	1093	l r side breast	0.45011583	-2.330393553	0.4018242657184601	3.618605613708496	-2.305247784	-4.086119175
17	1	1094	l r side breast	0.5997675657272339	-1.930781722	0.5088262557983398	2.8724164962768555	-2.720128536	-5.177294731
18	1	1131	l r side breast	-1.314069152	-8.756925583	-1.067542076	0.7413074374198914	-0.34604609	-6.603651524
19	1	1132	l r side breast	0.3391876518726349	-5.491604328	0.082907654	0.14188827574253082	0.3633917272090912	-3.768157959
20	1	1133	l r side breast	0.8698070049285889	-4.329964161	0.19550077617168427	3.168471097946167	-1.117723823	-4.914712906

Figure 12: A sample of the csv file showing the image index, the patch index, one class label and some features

4.4 Random Forest Classifier implementation

An extra idea to get better results was to add a supervised classifier to an unsupervised classifier. So, by initializing the K to a small value I was able to extract from the K-means algorithm the breast area and apply Random Forest Classifier to those patches. I also use the csv file I created to help me train my model and add labels. So, I extract the features of the patches from the csv and split them randomly to 80% for training and 20% for testing and then I numerus the labels to be easier to work with and I train the model using this data for 100 epoch. I also extracted the features of the breast area patches that I got from the K-means algorithm. Applying the trained model as it is on these data did not work because the model was clustering all the breast area patches instead of the cancer area only, so I had to add a probability threshold to use for the assigning of patches to a cluster. I set the threshold to 0.6 or 0.7 and extract the predicted probability for each class for every patch and then I get from them the highest probability for each patch across all classes and then I use the threshold to filter the patches with a classifier's confidence level. Lastly, I initialize the new clusters and fill them with the patches that pass the threshold in a shape: (img_idx, patch_idx).

4.5 K-means implementation with additional images

For the validation of my results, I also needed to see how my model will react with thermographs of healthy women as it should not find any anomalies. At first, I thought that I could just run the k-means implementation using the features from the images with the healthy women, but that approach was wrong because, the model had no cancer area to compare with the other areas. The right approach was to train a model using features from sick women and then pass from the model features from images with healthy women and see the difference.

Algorithm 3: K-means implementation with additional images

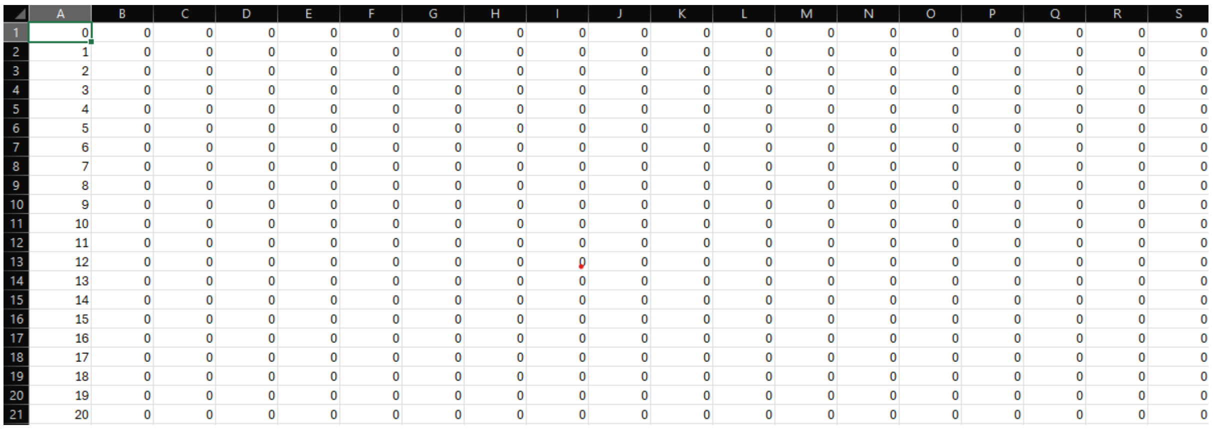
```
    Data: Patch-level features from DINOv2 for the extra images
    Result: Visualization of the cluster on the new images
 1  Apply the k-means implementation with features from images with sick women
 2  Define the path to a new image .pt file (e.g., "dinov2_features_healthy_10_pca.pt")
 3  Load the tensor using torch.load()
 4  Determine the number of new images:
 5  For img_idx from 0 to N_NEW_IMGS-1:
 6      Extract features for the current image
 7      (Optional) Apply dimensionality reduction using PCA:
        new_image_features_reduced ← pca.transform(image_features)
 8      Normalize features using previously fitted scaler:
        new_image_features_norm ← scaler.transform(image_features)
 9      Predict cluster labels using pre-trained Kmeans:
        new_image_clusters ← kmeans.predict(new_image_features_norm)
10      Reshape cluster assignments back to 2D image grid:
        new_image_clusters.shape ← (PATCH_ROWS, PATCH_COLS) or
        (N_NEW_IMGS, NUM_PATCHES)
11  end
```

4.6 Labelling for evaluation

As I did not had labels for my data, I had to think of a way to evaluate my model's result. A good idea to evaluate my results was to create a CSV file with N rows and P+1 columns where N is the number of images and P is the number of patches per image and the first column is the index of the image and I assign the value 1 to the patches that were in the area of the cancer and 0 to the patches that were not [Figure 13].

Algorithm 4: Labelling for evaluation

```
     Data: Number of columns for each row with 1's,the number each column with 1's
     Result: CSV file with 0's and 1's labels
1    Open a new CSV file for writing
2    For each row (from 1 to rows):
3        Print which row is currently being filled
4        Ask the user: "How many 1's do you want in this row?"
5        Ask user to enter count column numbers (1 to cols), separated by spaces
6        Convert input to a list of integers
7        Validate: Input length equals count
8        Validate: All indices are within bounds (1 to cols)
9        If input is invalid:
10           Print error message and re-prompt
11       end
12       Start with a list: [row_index] + [0]*cols
13       Set positions at selected indices to 1 //index 1 maps to first column (data starts from
         column 1, index 1)
14       Write the row to the CSV file
15   end
```



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
12	11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
13	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
16	15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	18	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21	20	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Figure 13: Example of the CSV file with 21 images and the first 18 patches

4.7 Visualization

For analysing my results, I also had to see how my model was clustering the images. The fist approach was to show a sample of 5 patches per cluster but that was not helpful because the patches were so small, and I could not understand what each patch was showing [Figure 14].

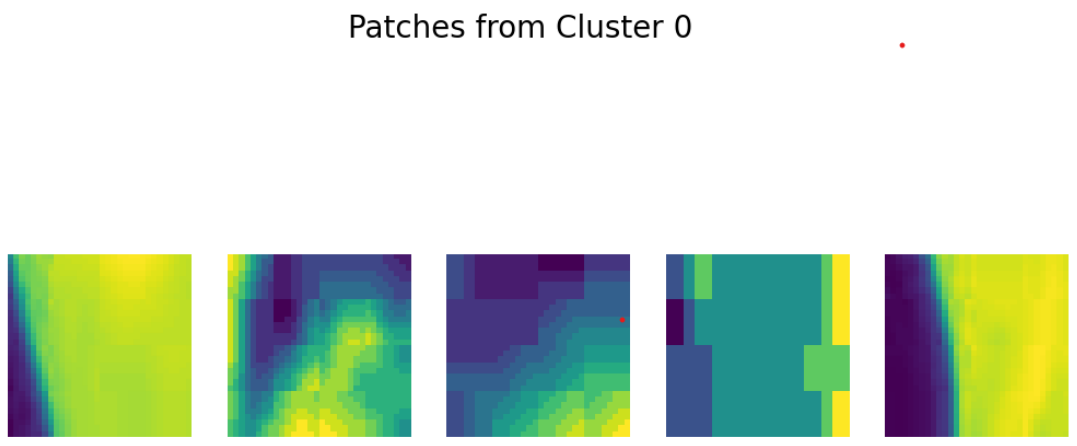


Figure 14: Visualization of the 5 patches of the first cluster

In the second approach I was creating a k*k heatmap for each cluster combining all the images where k is the number of the rows and columns of the patches and the heatmap was darker in the areas that multiple images had patches that belonged in the same cluster. This was better from the first approach but still had some problems. I was able to understand the average area that each cluster was taking but I could not easily compare each cluster because they were presented in deferent heatmaps, and I could not analyse each image alone because they were all combined in each heatmap [Figure 15].

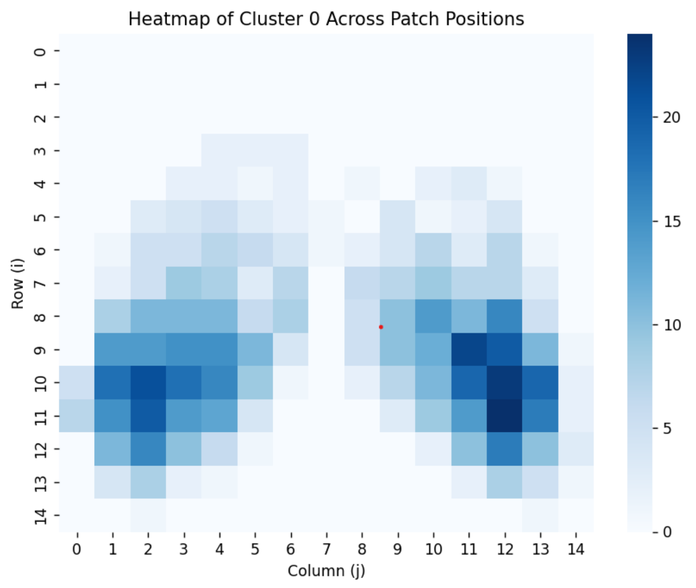


Figure 15: Visualization using a heatmap of the patches from all the images that belongs in the first cluster using 25 clusters and 15X15 patches

So, for the last approach I took the heatmap idea and change it to fix the problem I mentioned. This time I created a k*k heatmap for each image where it was combining all the clusters with different colour for each cluster and this way, I was able to analyse each image, and it was easier to see the exact regions for each cluster. I also put the number of the cluster in each patch because some colures were not so different [Figure 16]. Lastly to be able to compare the heatmaps with the original images I created a diagram on the images showing the patches and their indexes (i , j) [Figure 17].

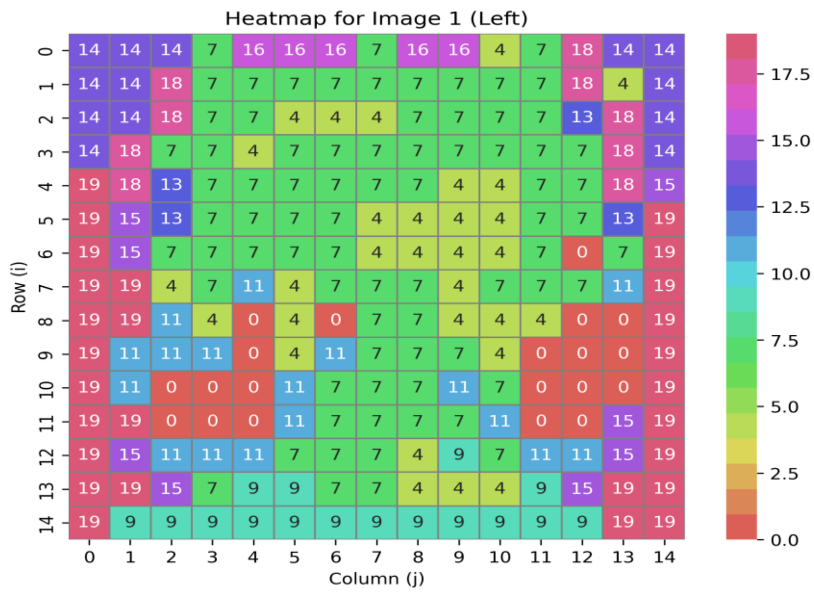


Figure 16: Visualization using a heatmap of the image 1 using 20 clusters and 15*15 patches and showing the cluster number of each patch

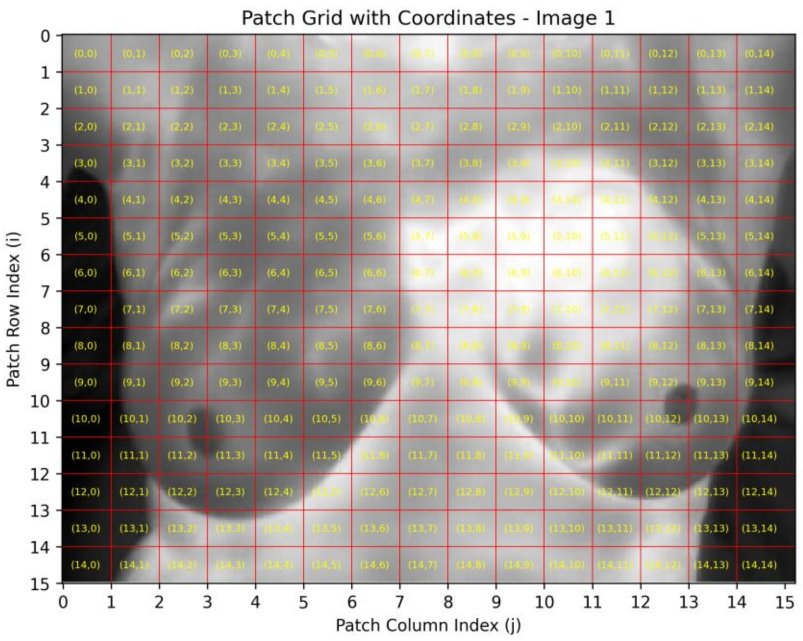


Figure 17: Diagram on image 1 showing the 15*15 patches and the indexes of each patch

Chapter 5

Experiments, Results & Discussion

5.1 Experiments

In my experiments I tried a lot of approaches and parameters so that I can maximize my metrics and more specifically the F1 score as is more important to show that the algorithm is not showing more ill patches from the actual ill patches.

5.1.1 Feature Map 15X15

I used a 15X15 feature map because I needed to try with a small number first and work my way up but 10X10 was too small and I did not get enough information for the clustering. Also, I tried each experiment with 20 ,25 and 30 clusters to report the difference and find the best cluster number for each case.

5.1.1.1 Raw Features

At first, I tried to just cluster the patches without any preprocessing, and the clusters with the tumour area are 1 and 8 for 20 clusters, 2 for 25 and 14 for 30 clusters .

5.1.1.2 PCA Feature reduction to 128

My data from preprocessing using dinov2 were produced with 384 features that may have been a lot for the classifier, and it may cause the model to overfit and not generalize. So, I tried to reduce the features using PCA to 128 because they are less than the original 384 features but not that few to cause the model to have poor training. With this new data the tumour clusters now are 18 for 20 clusters , 2 for 25 clusters and 0 and 22 for 30 clusters

5.1.1.3 Tumour cluster reassign using K-means centres

I notice from the previous results that the cluster with the tumour area in most of the images was detecting the tumour, but it was too general, so I thought I can reassign some of the patches from the tumour cluster with a different cluster and first I tried to use the k-means centre of the cluster to reassign the furthest patches. Also I set the number of patches that I wanted to change based on the cluster’s quantity, to be more specific I ignore the images with less than 10 patches, I change the 1/4 of the furthest patches in the tumour cluster if they are less than 15, I change the 1/3 if they are less than 20 and if they are more I change the half of the furthest patches in the tumour cluster.

5.1.1.3.1 Raw Features with tumour cluster reassign using K-means centres

I applied cluster reassign using the k-means centre on the raw data. Again, the clusters with the tumour area are 1 and 8 for 20 clusters, 2 for 25 and 14 for 30 clusters.

5.1.1.3.2 PCA Feature reduction to 128 with cluster reassign using K-means centres

I applied cluster reassign using the k-means centre on the data with pca features reduction to 128. Like previous [5.1.1.2] with feature reduction the tumour clusters are 18 for 20 clusters, 2 for 25 clusters and 0 and 22 for 30 clusters.

5.1.1.4 Tumour cluster reassign using Euclidean distance

Using the heatmap that I created with the experiment 5.1.1.3 ,where I reassigned the tumour clusters using the k-means centre of the clusters I observed that the centre was not physically in the centre of the cluster patches and in some images most of the patches in the tumour cluster were close or in the tumour area. This is why I thought to compute the physical cluster centre by my self using the Euclidean distance between the patches and use that centres to determine which patches are the furthest. And then for the number of the changed patches I followed the same pipeline as previous [5.1.1.3] .

5.1.1.4.1 Raw Features with tumour cluster reassign using Euclidean distance

I applied cluster reassign using the Euclidean distance for finding the centre on the raw data. The tumour clusters are the same with the raw features without clusters reassign.

5.1.1.4.2 PCA Feature reduction to 128 with tumour cluster reassign using Euclidean

I applied cluster reassign using the Euclidean distance for finding the centre on the data with pca features reduction to 128. Once again, the tumour clusters are the same with the experiment with pca reduction without clusters reassign.

5.1.1.5 Using more data

In another experiment I tried to train the model with more data because the low results might be because of the lack of big dataset. So, I tried to use both “Breast thermography” dataset that I have been using and the “A New Database for Breast Research with Infrared Image” dataset but only evaluate on the 35 images of “Breast thermography” dataset.

5.1.1.5.1 Using more data with raw features

I added more data and used raw features. The tumour clusters are 12 for 20 clusters, 9 for 25 clusters and 4 for 30 clusters.

5.1.1.5.2 Using more data with PCA Feature reduction to 128

I applied pca features reduction to 128 to all the data. The tumour clusters are 18 for 20 clusters, 23 for 25 clusters and 12 for 30 clusters.

5.1.1.6 Patch selection

Another idea to make the algorithm more precise than it is originally, is to exclude some patches from the images that were not important in any of the images before the clustering and that made the images more specific, and we ignored unnecessary features. The patches that I select to exclude were from 0 to 45 that were mostly the nerk and the arms area and 209 to 224 that were mostly the belly area.

5.1.1.6.1 Raw features with patch selection

I applied the patch selection on the data with raw features. The clusters with the tumour area are 3 for 20 clusters, 24 for 25 and 3 for 30 clusters.

5.1.1.6.2 PCA Feature reduction to 128 with patch selection

I applied the patch selection on the data with pca features reduction to 128. The clusters with the tumour area are 18 for 20 clusters, 1 for 25 and 2 for 30 clusters.

5.1.1.7 Patch selection and tumour cluster reassign using Euclidean

Since I had some improvement with patch selection and tumour cluster reassign using Euclidean which was better than using the K-means centres I decided to try combining the two ideas and analyse the results.

5.1.2 Feature Map 25X25

I tried a lot of variation of 15X15 patches, so I had to also try different patch map to see how my model works with more patches, and I tried 25X25 and analyse the results with some of the ideas I tried with the previous experiment but with more clusters.

5.1.2.1 Raw features

First, I tried the model using data with raw features with 50 clusters only because with less or more the model was giving very bad results and the tumour cluster is 26.

5.1.2.2 PCA Feature reduction to 128

Then, I tried adding feature reduction to 128 on the data and tried the model with 30 and 50 clusters and the tumour clusters are 29 for 30 clusters and 6 for 50 clusters.

5.1.2.3 Tumour cluster reassign using Euclidean distance

As I realised from the 15x15 experiments I had better results by reassigning the tumour clusters using the Euclidean distance for the detection of the physical cluster centres, so I did not try using k-means centres for this experiment. Also, the data with reduced features using 30 clusters gave very bad results with this idea so I kept only the 50 cluster trials. For the raw features the tumour cluster is 26 and for the reduced features is 6.

5.1.2.4 Patch selection

Lastly, I tried the idea of selecting some patches for the classification to reduce the extra information. I extracted the patches 0-150 and 550-625. The tumour clusters are 27 for raw features and 0 for reduced features.

5.1.3 Testing trained model on healthy women

I also tried to train a model on women with breast cancer and pass through the trained model denov2 features with the same processing from some images with healthy women to analyse the results and see if the tumour area cluster appears on the healthy images. Because I had the best results in the experiment with raw features, patch reassign using Euclidean distance, patch selection and 30 cluster [Graph 11] and with reduced features to 128, patch reassign and 20 clusters [Graph 6], so I pass 10 new images from those models and see the new heatmaps

5.1.4 Supervised on unsupervised training

Lastly, I tried to add supervised training on unsupervised training, and I used Random Forest classifier and the labels I created with 40X40 patches and 348 features with the "Breast Thermography" dataset and applied 10 images from the "A New Database for Breast Research with Infrared Image" dataset [7].

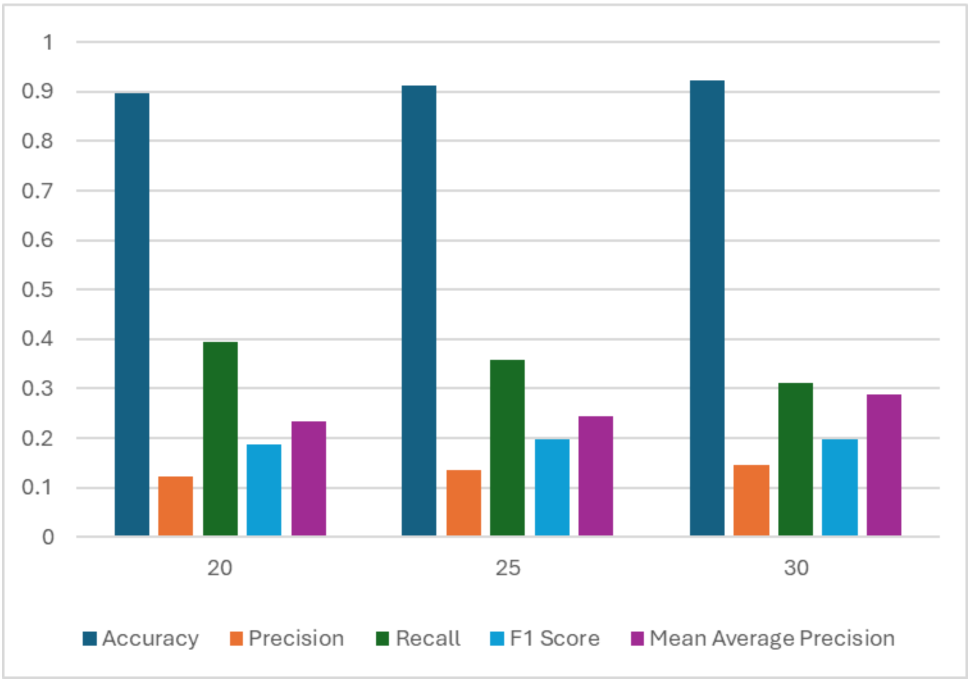
5.2 Results & Discussion

For analysing the results of the experiments, I used the confusion matrix with the ground truths I already mention. More specific I did not give attention to the accuracy because of the imbalance of the TP and TN as the TN were way more and the model even with non TN was succeeding more than 80% accuracy. I focused only on precision ,recall and f1 score but more on the precision and f1 score because I want the model to be precise and I want to minimize the FP as much as possible. Also, I used the metric mean average precision with IoU which is a common way to analyse models with object detection like tumour detection. Moreover, I created a baseline for comparison using a temperature threshold of 85 and my model in each experiment had much better results. For each I will display a plot with the best runs of the model for that experiment compering them with the baseline and the ground truths for better understanding.

5.2.1 Feature Map 15X15

5.2.1.1 Raw Features

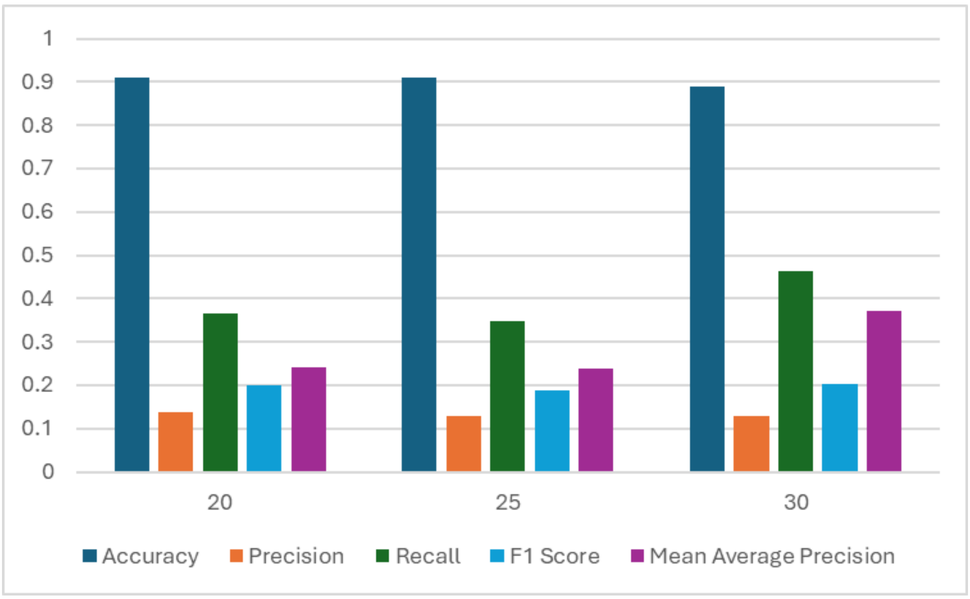
We can see in the chart [Graph 1] with 20 clusters we have the lowest precision, mean AP and f1 score with 12%, 24% and 19% respectively and it increases with the increase of the clusters peaking up to 15%, 29% and 20% respectively with 30 clusters. But we can see that recall is decreasing from 40% to 31% because with more clusters the cluster with the tumour area has less patches.



Graph 1:Raw data

5.2.1.2 PCA Feature reduction to 128

We can see in the graph [Graph 2] that now the precision is decreasing as the clusters increase from 14% with 20 clusters to 13% with 30 clusters. The recall and the mean AP peaks to 46% and 41% respectively with 30 clusters and the lowest are 35% and 37% with 25 clusters and lastly the f1 score is almost the same with 20 and 30 clusters with 20% but with 25 clusters it slightly decreases to 19%.

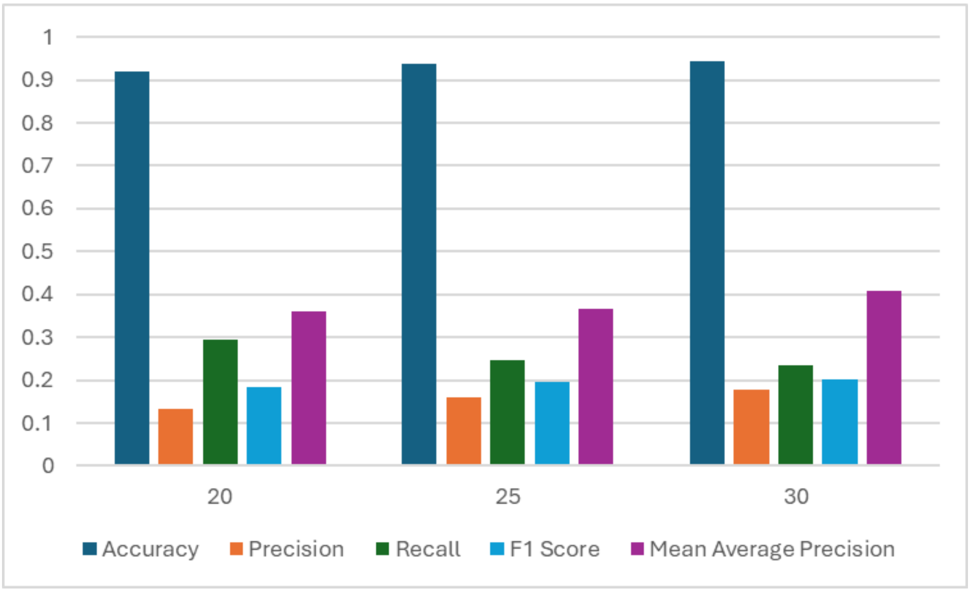


Graph 2: PCA Feature reduction to 128

5.2.1.3 Tumour cluster reassign using K-means centres

5.2.1.3.1 Raw Features with tumour cluster reassign using K-means centres

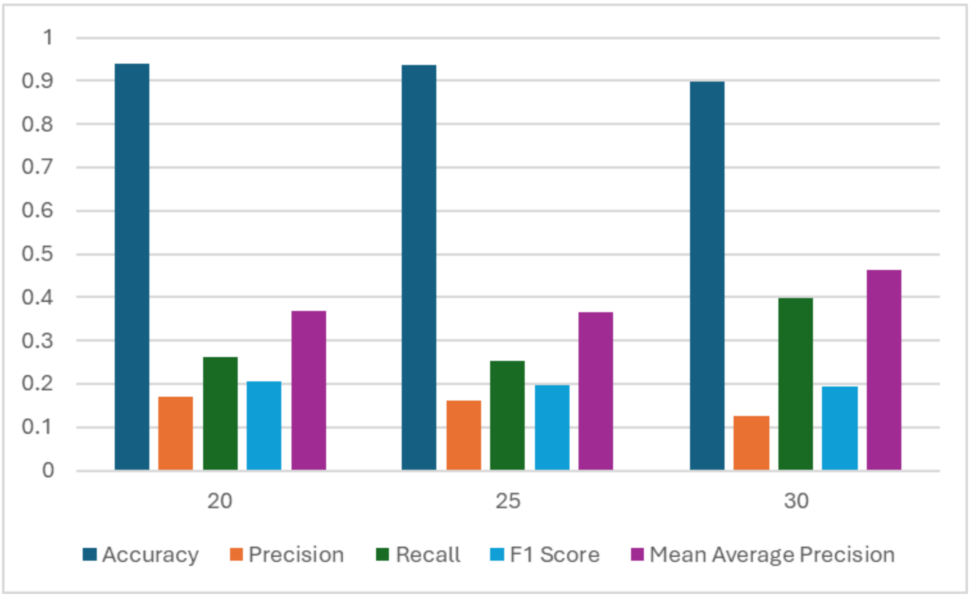
As we can see in the chart [Graph 3] that our result has the same pattern and they are around the same with the previous results [Graph 1] but now with 20 clusters the precision starts from 13% and the f1 score from 18% and they get to 18% and 20% respectively with 30 clusters so we can see that now we are more precise. But the mean AP has increased to 41% with 30 clusters. We also notice that because specified more the tumour cluster the recall starts from 29% with 20 clusters and gets to 24% with 30 clusters.



Graph 3:Raw Features with tumour cluster reassign using K-means centres

5.2.1.3.2 PCA Feature reduction to 128 with cluster reassign using K-means centres

The results have also the same pattern as previous [Graph 2] but now the peak precision and f1 score are higher with 17% and 21% respectively with 20 clusters and mean AP is 46% with 30 clusters but, peak recall is lower with 40% using 30 clusters [Graph 4].

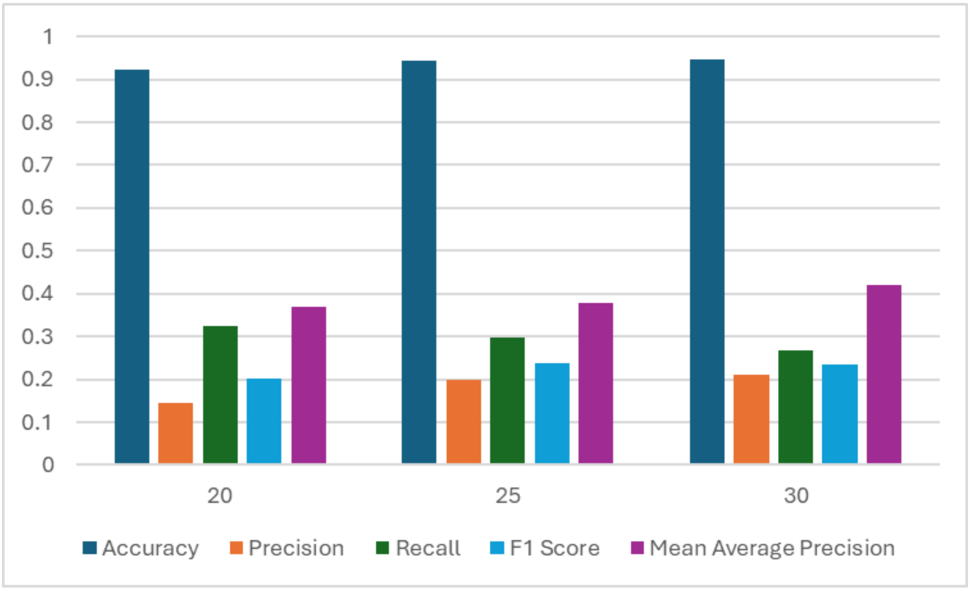


Graph 4: PCA Feature reduction to 128 with cluster reassign using K-means centres

5.2.1.4 Tumour cluster reassign using Euclidean distance

5.2.1.4.1 Raw Features with tumour cluster reassign using Euclidean distance

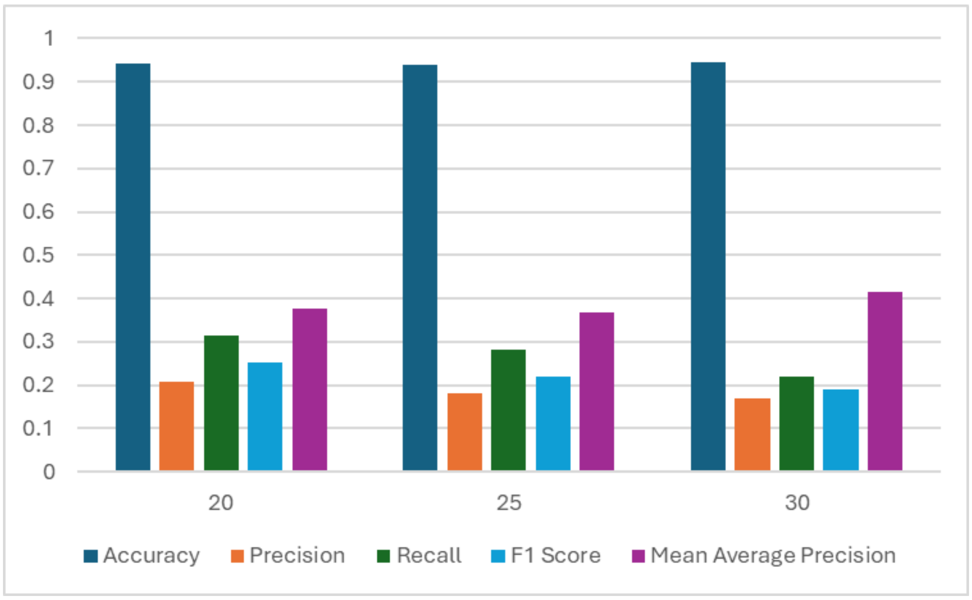
As I expected the result are better from using the k-means centres but unfortunately the improvement is minimum. The precision start from 15% with 20 clusters and peaks to 21% with 30 cluster and we have +3% improvement from the k-means but the mean AP only increased 1% with 42%. Also, the f1 score peaks to 24% with 25 clusters with +4% improvement. Lastly recall peaked to 32% with 20 clusters with +3% improvement [Graph 5].



Graph 5: Raw Features with tumour cluster reassign using Euclidean distance

5.2.1.4.2 PCA Feature reduction to 128 with tumour cluster reassign using Euclidean

Like the raw data the result with this idea is better than using the k-means centres and they are the best results yet with f1 score peaking to 25% and precision peaking to 21% with +4% improvement in both metrics using 20 clusters but, the mean AP is 38% in that case and it peaked to 41% with 30 clusters. Unfortunately, the recall was decreased to 32% and as the clusters increased the metrics were decreasing [Graph 6].

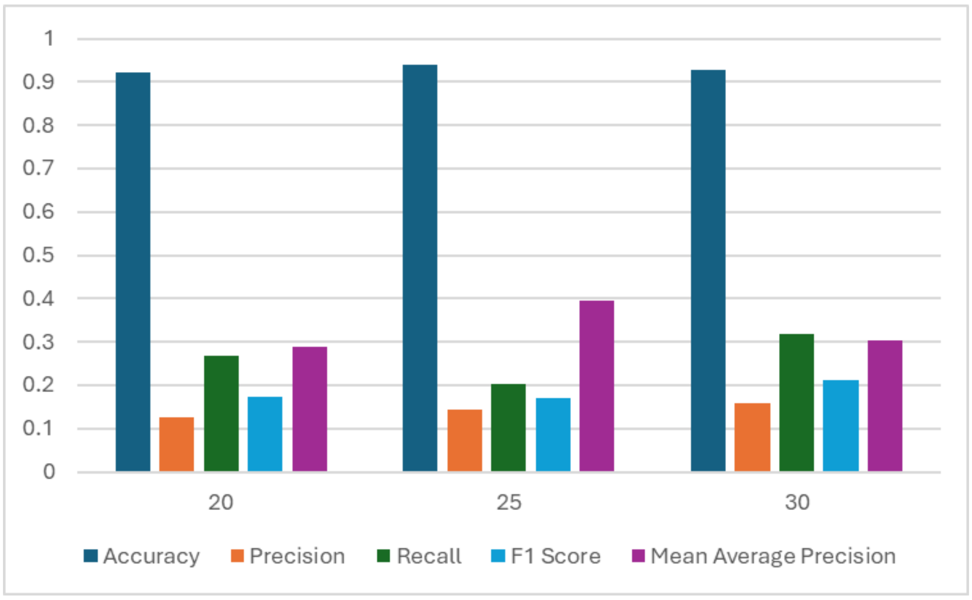


Graph 6: PCA Feature reduction to 128 with tumour cluster reassign using Euclidean

5.2.1.5 Using more data

5.2.1.5.1 Using more data with raw features

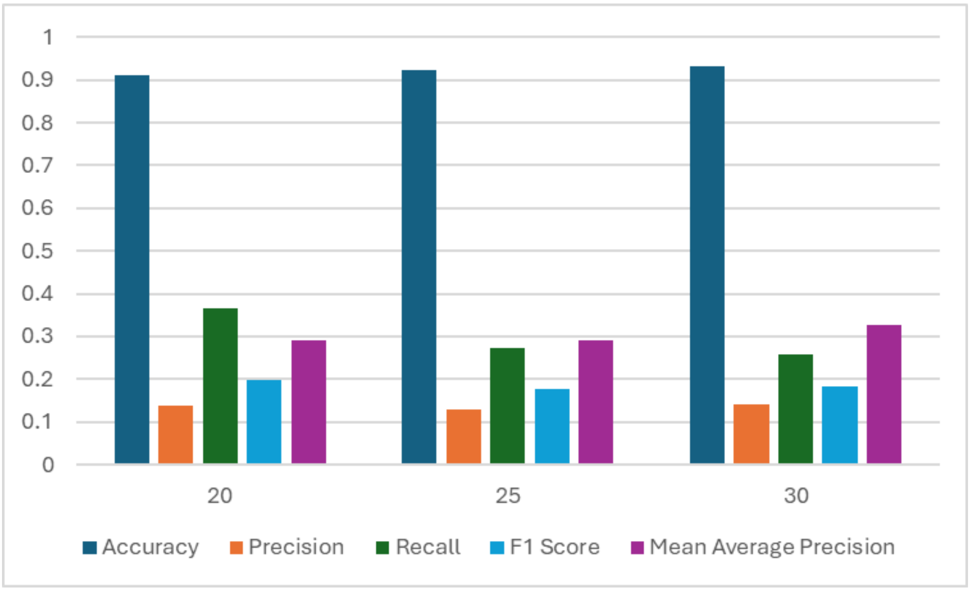
The results [Graph 7] showed that with more data the model has a very minimum improvement from the training with only the "Breast thermography" dataset as with 30 clusters precision is 16% and f1 score is still 20% and with 20 clusters the recall peaked with only 27%. Also the mean AP peaked to 40% with 25 clusters.



Graph 7: Using more data with raw features

5.2.1.5.2 Using more data with PCA Feature reduction to 128

Unfortunately, we have almost the same results with the training with only the one dataset with peak precision 14% peak f1 score 20% and peak recall 37% but, the mean AP is less with 33%. So, more data was not helpful with reduced data [Graph 8].

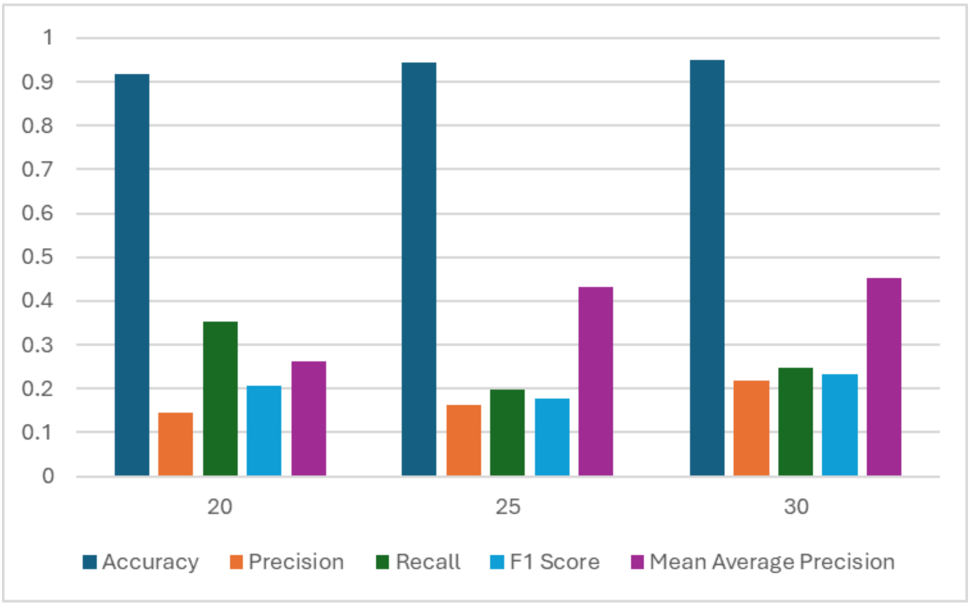


Graph 8: Using more data with PCA Feature reduction to 128

5.2.1.6 Patch selection

5.2.1.6.1 Raw features with patch selection

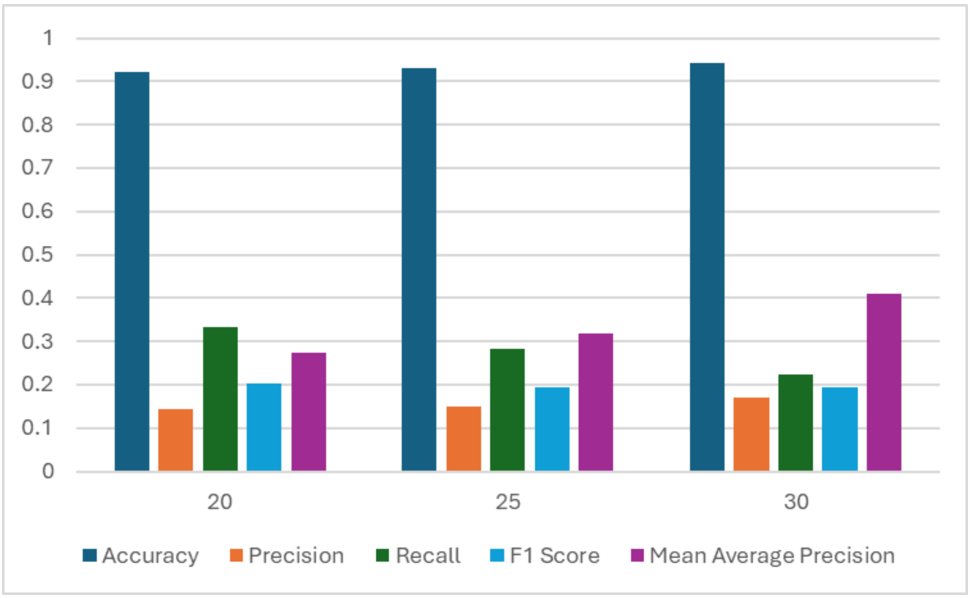
With this method I specified the images and managed to get a good improvement on precision with peak 22% with 30 clusters as well as on mean AP with 45%. Also, f1 is better with 23% but the recall again is 35% with 20 clusters [Graph 9].



Graph 9: Raw features with patch selection

5.2.1.6.2 PCA Feature reduction to 128 with patch selection

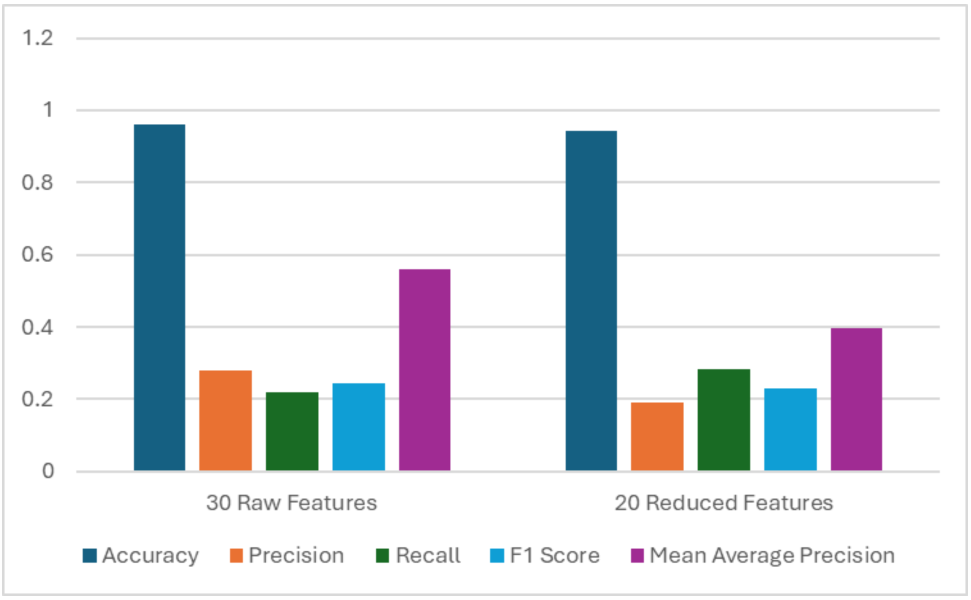
We can notice [Graph 10] that the precision is higher than the pca reduced data without patch selection with 17% using 30 clusters but the f1 score remains the same and the recall is lower with peak 33% but the mean AP peaked higher with 42%. On this case we can see that the results are better without PCA reduction.



Graph 10: PCA Feature reduction to 128 with patch selection

5.2.1.7 Patch selection and tumour cluster reassign using Euclidean

I tried the model only for 30 cluster for the data with the raw features because in most of the experiments the data with the raw features had better results from the other and also the experiment with patch selection on raw features had the best precision so far and I thought that adding cluster reassigning would improve it even more and it actually does with 28% and f1 score 25% and the best mean AP so far with 56% [Graph 11]. I also tried the 20 clusters with reduced features, because it had also the best results in most of the experiment but, unfortunately the metrics are better than the experiment with patch selection on reduced features but, they are not better than the one with the cluster reassign using Euclidian on reduced data and we see that the precision peaks to 19% ,the f1 score to 23% and the mean AP to 40% [Graph 11].

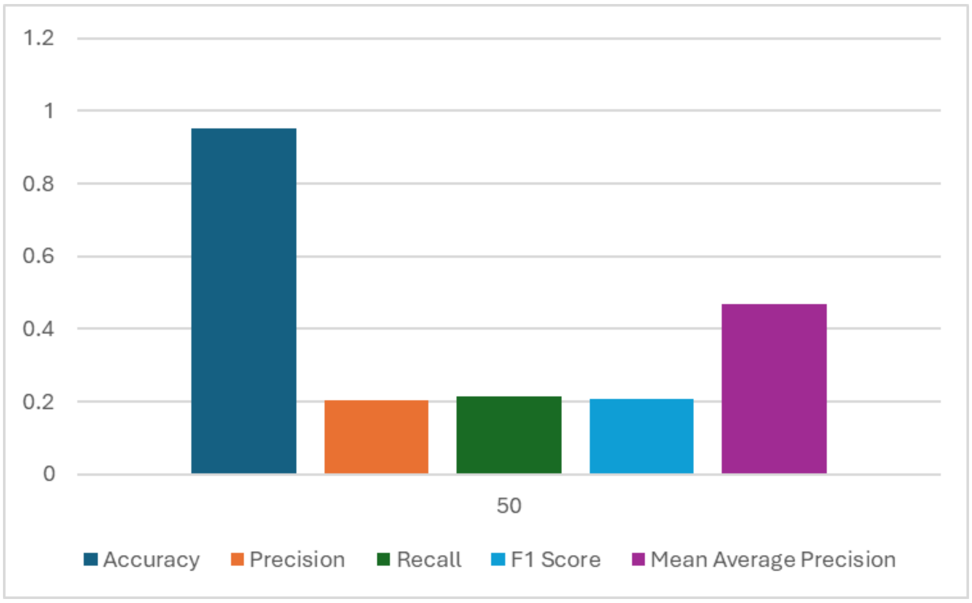


Graph 11:Patch selection and tumour cluster reassign using Euclidean

5.2.2 25X25

5.2.2.1 Raw features

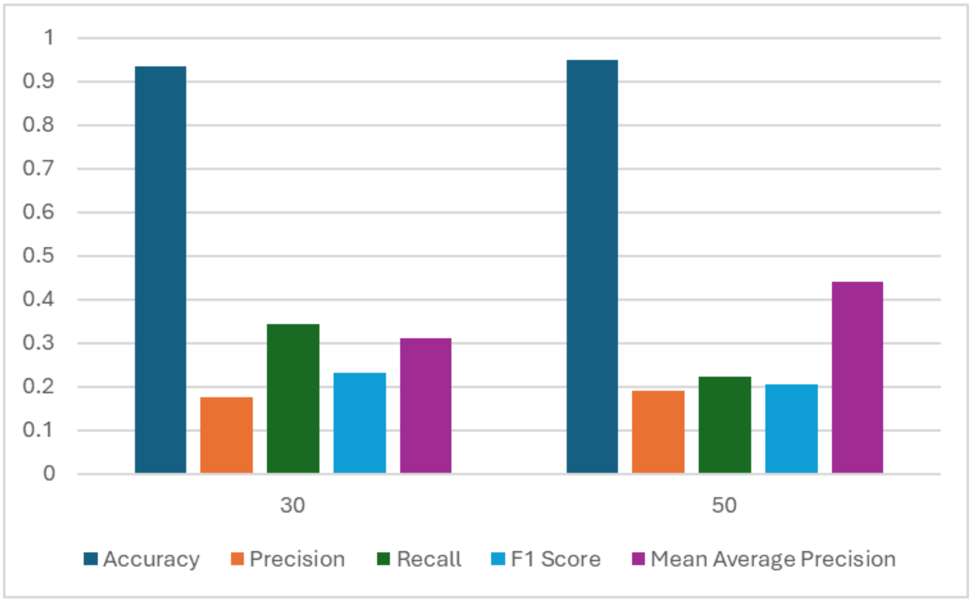
We can see that the model in comparison with the 15X15 [Graph 1] has better precision with 20% and slightly better f1 score with 21% but worst recall with 22% as well as mean AP with 47% [Graph 12].



Graph 12:Raw features

5.2.2.2 PCA Feature reduction to 128

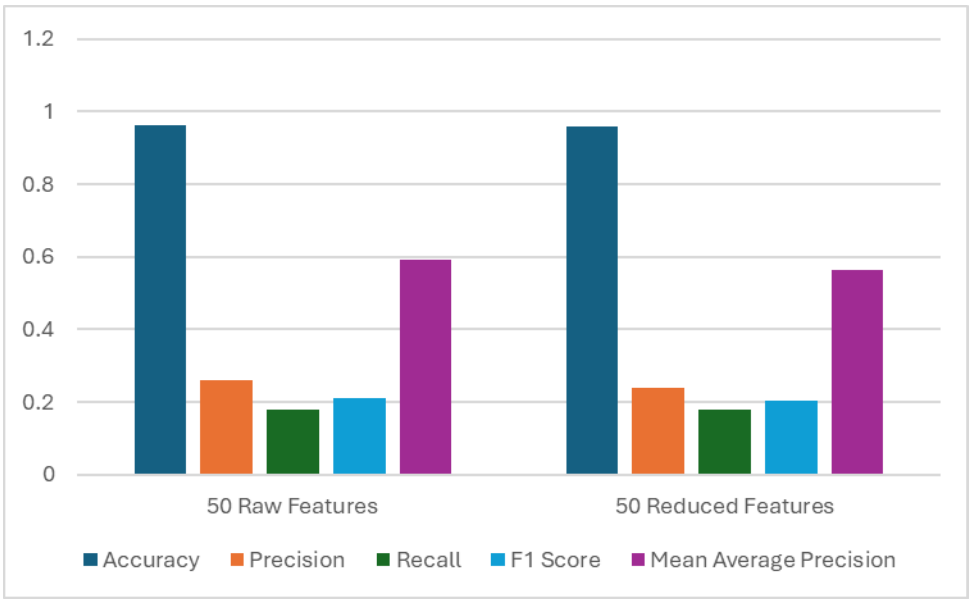
We can see that the model with 30 clusters has better recall which is 35% and is lower than the 15X15 experiment [Graph 2] but the precision and the f1 score is better with 19% and 23% respectively. With 50 clusters tho we have better mean AP with 44% [Graph 13].



Graph 12: PCA Feature reduction to 128

5.2.2.3 Tumour cluster reassign using Euclidean distance

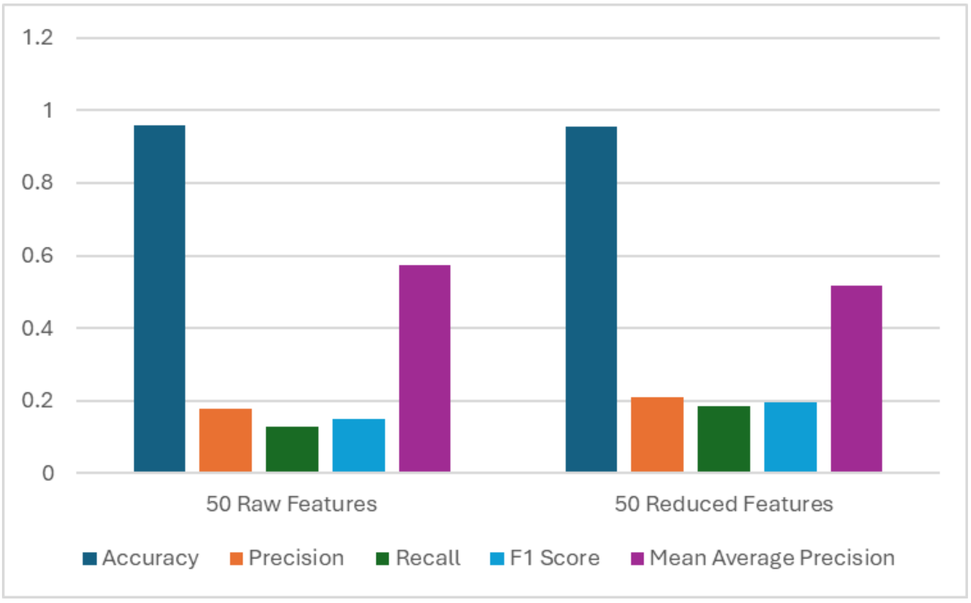
We observe from the graph [Graph 14] that using the clusters reassigning on both raw features and reduced features the model is more precise as the precision increased to 26% and 24% respectively and the mean AP increased to 59% and 56% respectively also, the f1 score remains the same with 21% in both trials, but in both trials the recall reduced to 18%.



Graph 13: Tumour cluster reassign using Euclidean distance

5.2.2.4 Patch selection

We can see that in this case with raw data the model is not working well as all the metrics are worst from the original result [Graph 12] with precision 18%, recall 13% and f1 score 15% [Chart 15]. With reduced features it has almost the same results with the original [Graph 13] with more precision 21% and less recall 19% but overall same f1 score around 20% [Graph 15]. But in both cases we have a good mean Ap with 57% and 52% respectively.

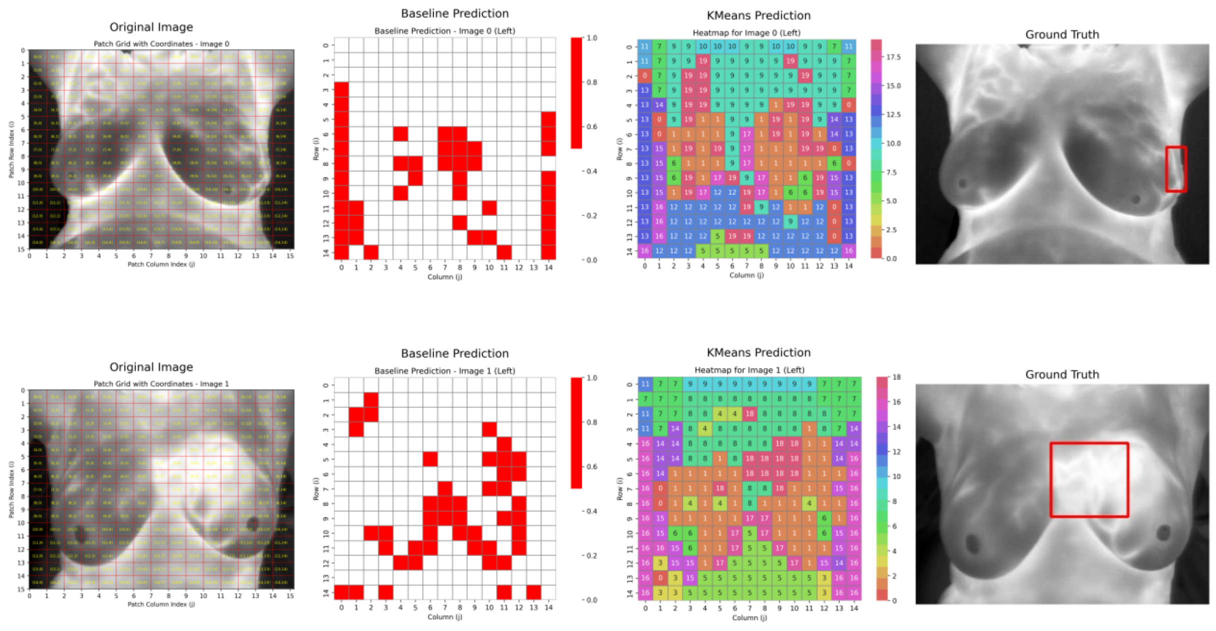


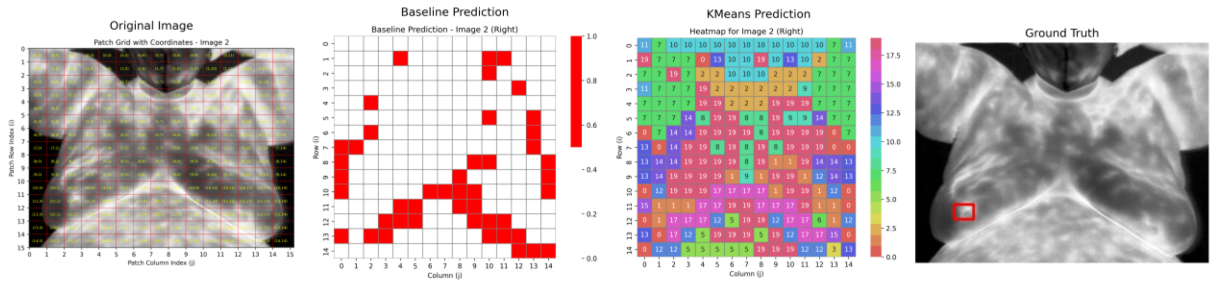
Graph 14: Patch selection

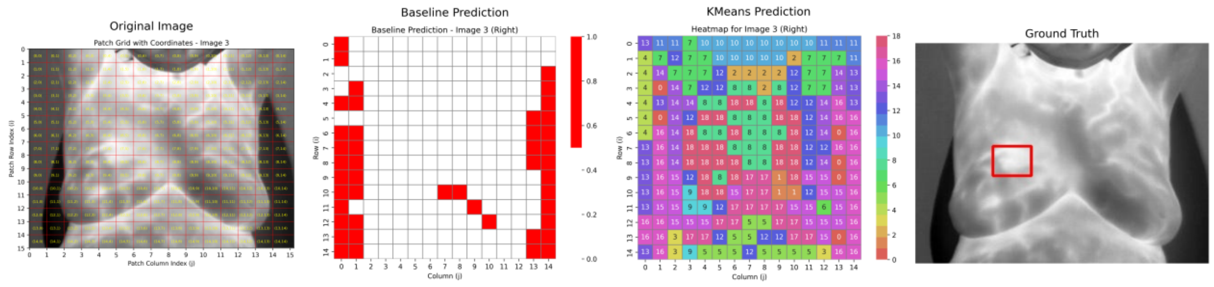
5.2.3 Visualisation and comparison with baseline

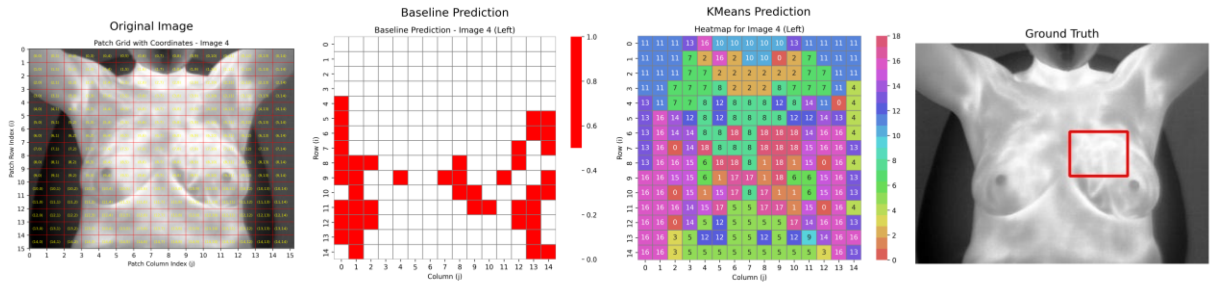
5.2.3.1 PCA Feature reduction to 128 with tumour cluster reassign using Euclidean

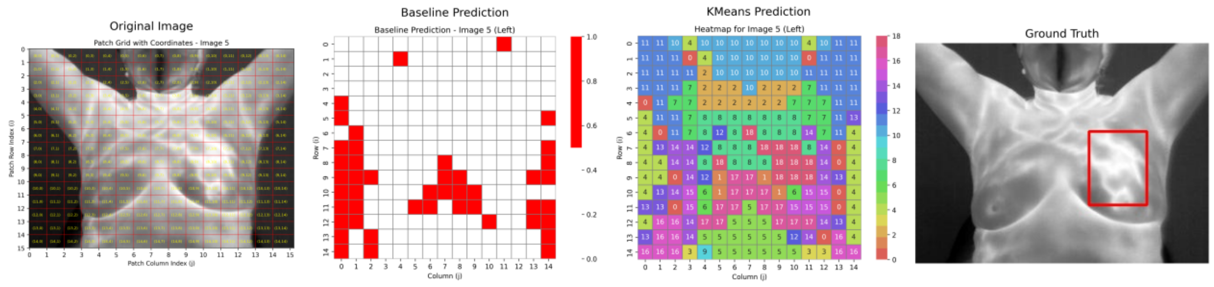
We can see the comparison with the experiment using 15X15 patches map, pca reduced features to 128, patches reassigning with Euclidean distance and 20 clusters and the tumour clusters is the 18 [Plot 1].

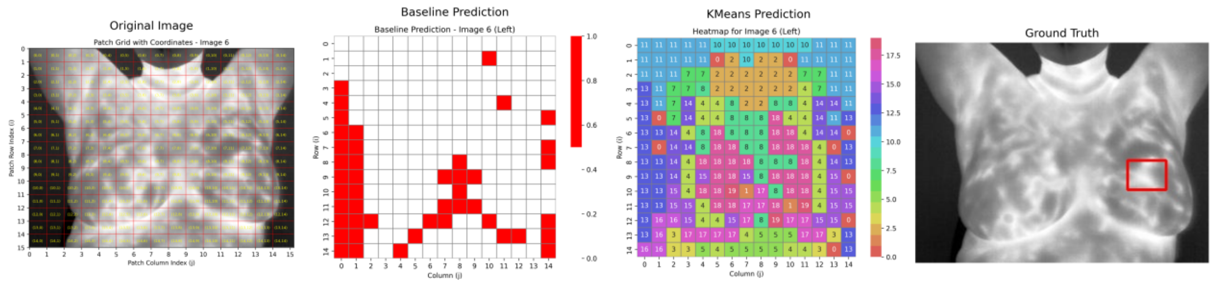

Original Image
Patch Grid with Coordinates - Image 0
Baseline Prediction
Baseline Prediction - Image 0 (Left)
KMeans Prediction
Heatmap for Image 0 (Left)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 1
Baseline Prediction
Baseline Prediction - Image 1 (Left)
KMeans Prediction
Heatmap for Image 1 (Left)
Ground Truth

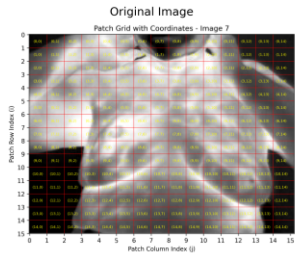
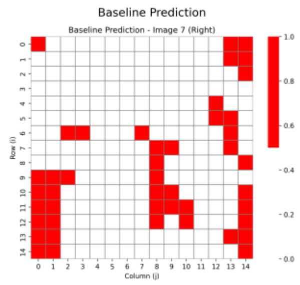
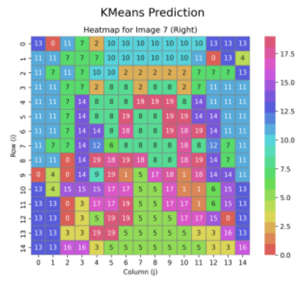
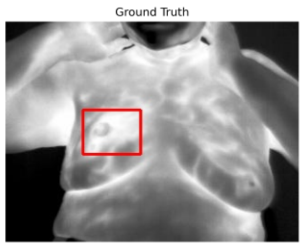
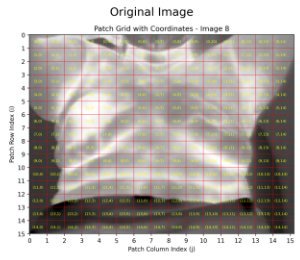
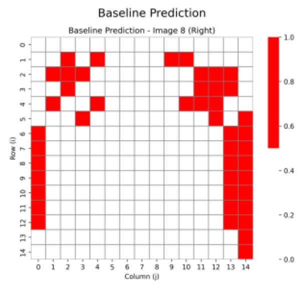
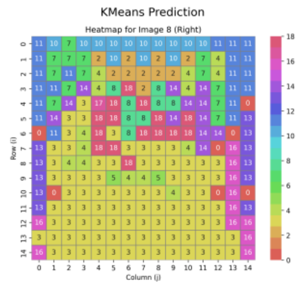
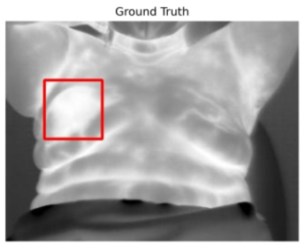
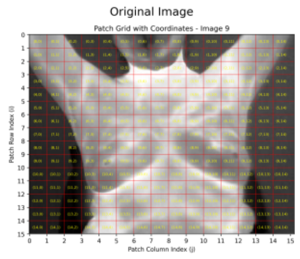
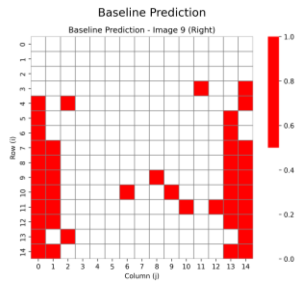
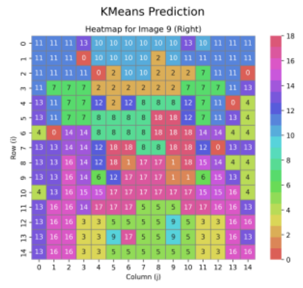
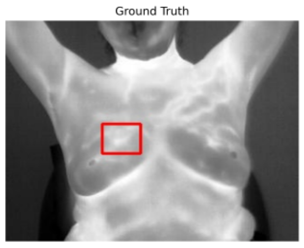
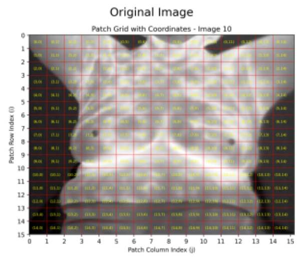
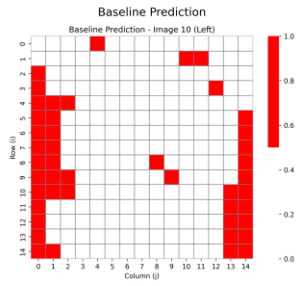
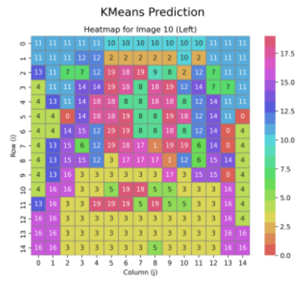
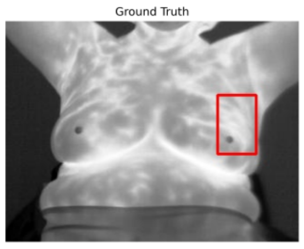
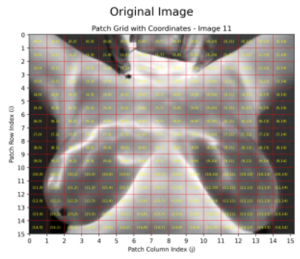
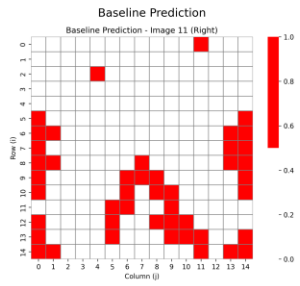
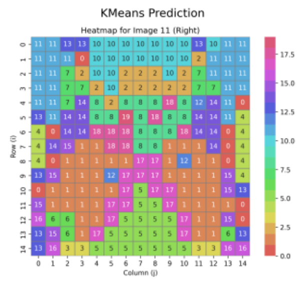
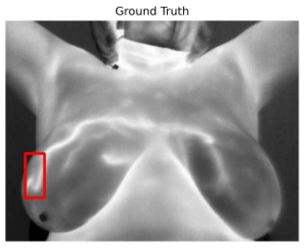

Original Image
Patch Grid with Coordinates - Image 2
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 2 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 2 (Right)
Row (i)
Column (j)
Ground Truth

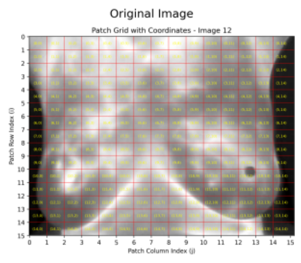

Original Image
Patch Grid with Coordinates - Image 3
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 3 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 3 (Right)
Row (i)
Column (j)
Ground Truth

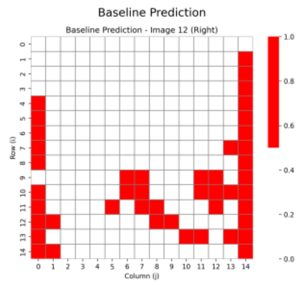

Original Image
Patch Grid with Coordinates - Image 4
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 4 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 4 (Left)
Row (i)
Column (j)
Ground Truth

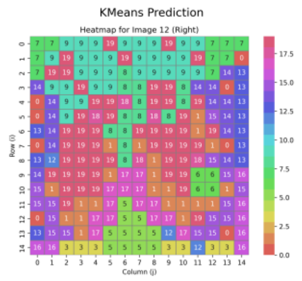

Original Image
Patch Grid with Coordinates - Image 5
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 5 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 5 (Left)
Row (i)
Column (j)
Ground Truth

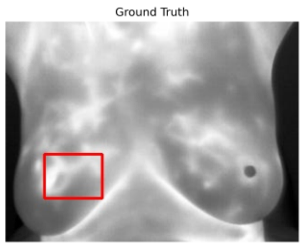

Original Image
Patch Grid with Coordinates - Image 6
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 6 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 6 (Left)
Row (i)
Column (j)
Ground Truth

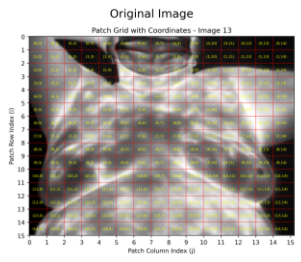

Original Image
Patch Grid with Coordinates - Image 7
Patch Row Index (i)
Patch Column Index (j)

Baseline Prediction
Baseline Prediction - Image 7 (Right)
Row (i)
Column (j)

KMeans Prediction
Heatmap for Image 7 (Right)
Row (i)
Column (j)

Ground Truth

Original Image
Patch Grid with Coordinates - Image 8
Patch Row Index (i)
Patch Column Index (j)

Baseline Prediction
Baseline Prediction - Image 8 (Right)
Row (i)
Column (j)

KMeans Prediction
Heatmap for Image 8 (Right)
Row (i)
Column (j)

Ground Truth

Original Image
Patch Grid with Coordinates - Image 9
Patch Row Index (i)
Patch Column Index (j)

Baseline Prediction
Baseline Prediction - Image 9 (Right)
Row (i)
Column (j)

KMeans Prediction
Heatmap for Image 9 (Right)
Row (i)
Column (j)

Ground Truth

Original Image
Patch Grid with Coordinates - Image 10
Patch Row Index (i)
Patch Column Index (j)

Baseline Prediction
Baseline Prediction - Image 10 (Left)
Row (i)
Column (j)

KMeans Prediction
Heatmap for Image 10 (Left)
Row (i)
Column (j)

Ground Truth

Original Image
Patch Grid with Coordinates - Image 11
Patch Row Index (i)
Patch Column Index (j)

Baseline Prediction
Baseline Prediction - Image 11 (Right)
Row (i)
Column (j)

KMeans Prediction
Heatmap for Image 11 (Right)
Row (i)
Column (j)

Ground Truth

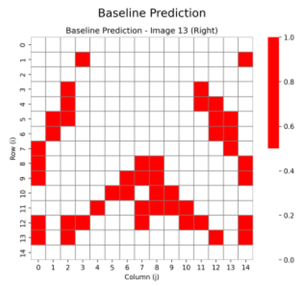

Original Image
Patch Grid with Coordinates - Image 12
Patch Row Index (i)
Patch Column Index (j)

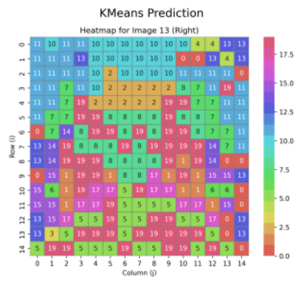

Baseline Prediction
Baseline Prediction - Image 12 (Right)
Row (i)
Column (j)

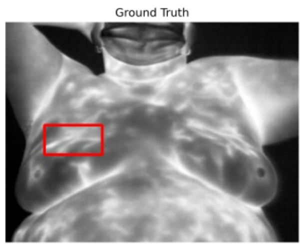

KMeans Prediction
Heatmap for Image 12 (Right)
Row (i)
Column (j)

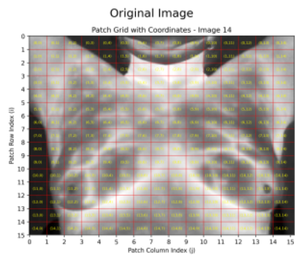

Ground Truth

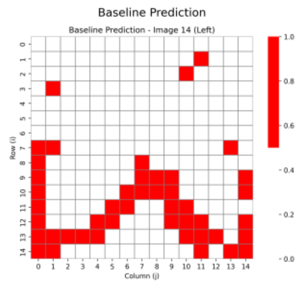

Original Image
Patch Grid with Coordinates - Image 13
Patch Row Index (i)
Patch Column Index (j)

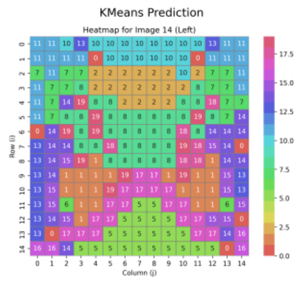

Baseline Prediction
Baseline Prediction - Image 13 (Right)
Row (i)
Column (j)

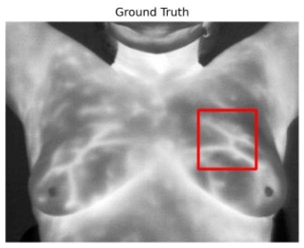

KMeans Prediction
Heatmap for Image 13 (Right)
Row (i)
Column (j)

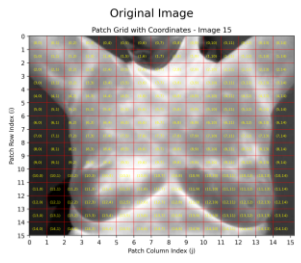

Ground Truth

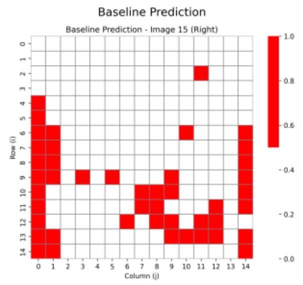

Original Image
Patch Grid with Coordinates - Image 14
Patch Row Index (i)
Patch Column Index (j)

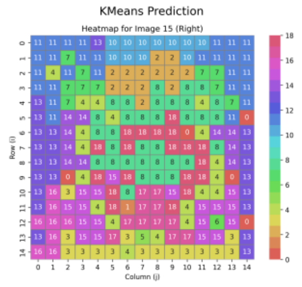

Baseline Prediction
Baseline Prediction - Image 14 (Left)
Row (i)
Column (j)

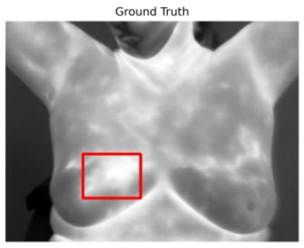

KMeans Prediction
Heatmap for Image 14 (Left)
Row (i)
Column (j)

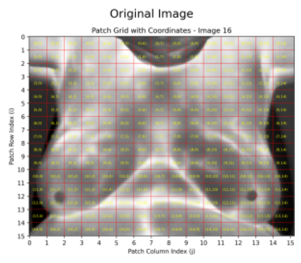

Ground Truth

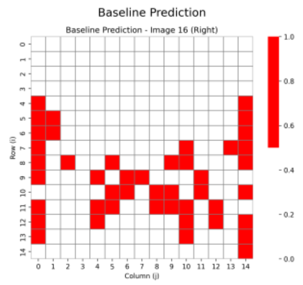

Original Image
Patch Grid with Coordinates - Image 15
Patch Row Index (i)
Patch Column Index (j)

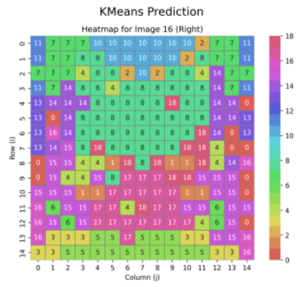

Baseline Prediction
Baseline Prediction - Image 15 (Right)
Row (i)
Column (j)

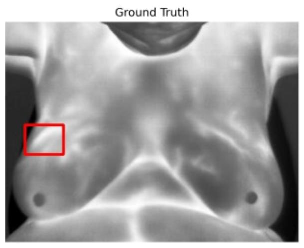

KMeans Prediction
Heatmap for Image 15 (Right)
Row (i)
Column (j)

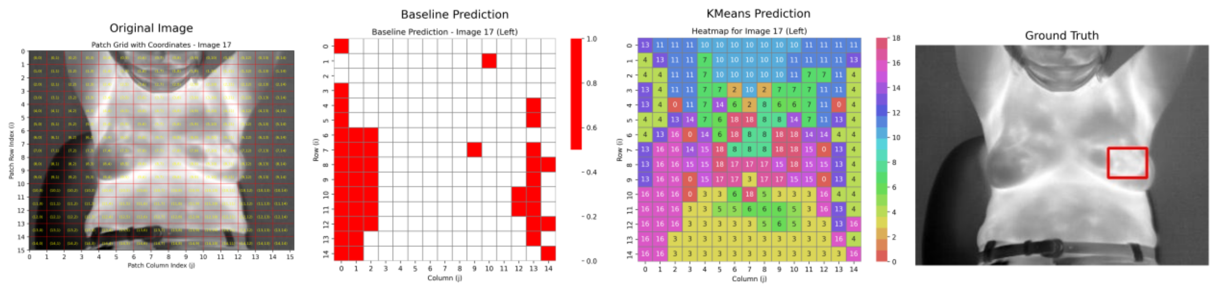

Ground Truth

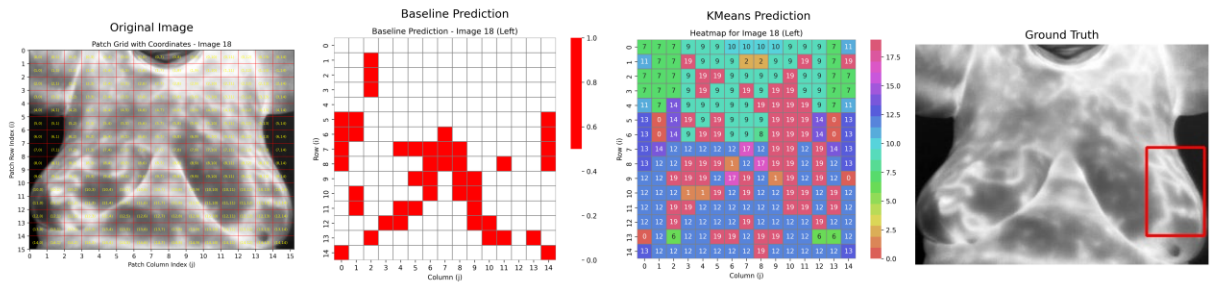

Original Image
Patch Grid with Coordinates - Image 16
Patch Row Index (i)
Patch Column Index (j)

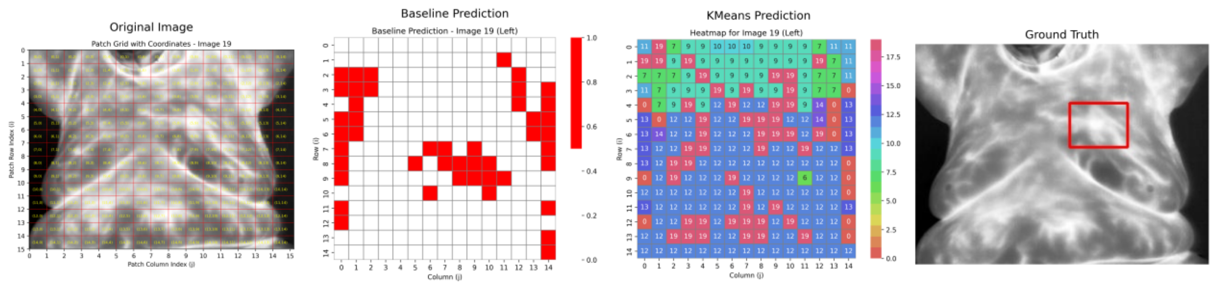

Baseline Prediction
Baseline Prediction - Image 16 (Right)
Row (i)
Column (j)

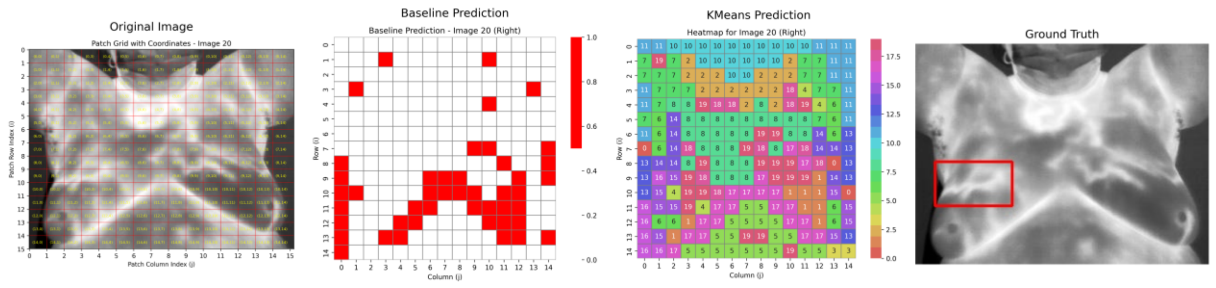

KMeans Prediction
Heatmap for Image 16 (Right)
Row (i)
Column (j)

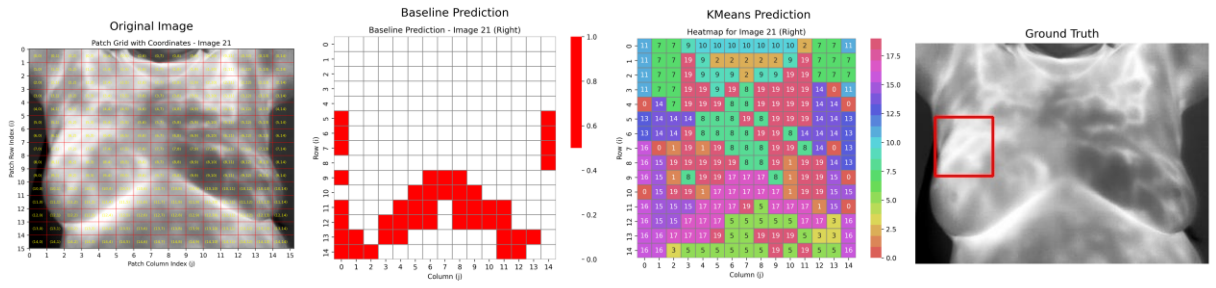

Ground Truth

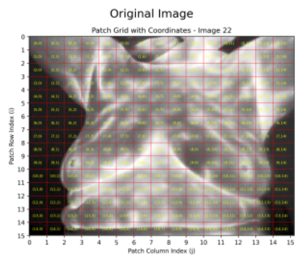

Original Image
Patch Grid with Coordinates - Image 17
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 17 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 17 (Left)
Row (i)
Column (j)
Ground Truth

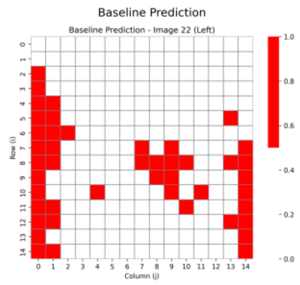

Original Image
Patch Grid with Coordinates - Image 18
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 18 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 18 (Left)
Row (i)
Column (j)
Ground Truth

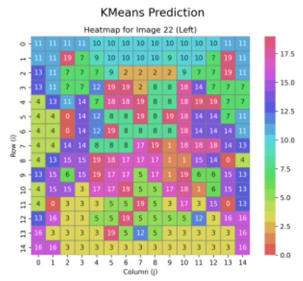

Original Image
Patch Grid with Coordinates - Image 19
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 19 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 19 (Left)
Row (i)
Column (j)
Ground Truth

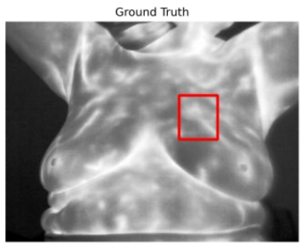

Original Image
Patch Grid with Coordinates - Image 20
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 20 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 20 (Right)
Row (i)
Column (j)
Ground Truth

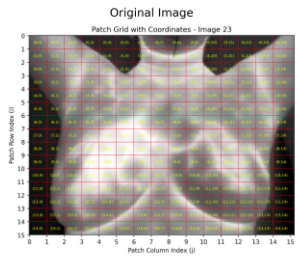

Original Image
Patch Grid with Coordinates - Image 21
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 21 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 21 (Right)
Row (i)
Column (j)
Ground Truth

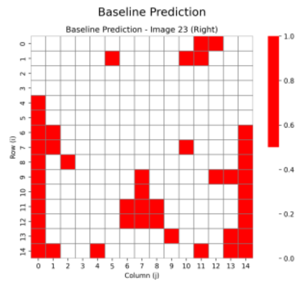

Original Image
Patch Grid with Coordinates - Image 22
Patch Row Index (i)
Patch Column Index (j)

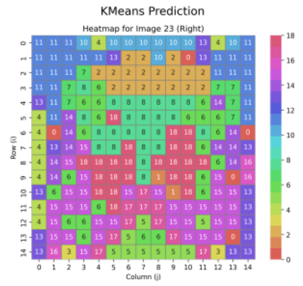

Baseline Prediction
Baseline Prediction - Image 22 (Left)
Row (i)
Column (j)

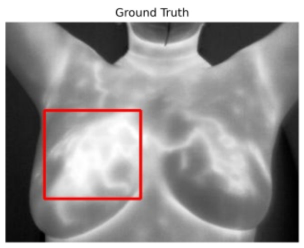

KMeans Prediction
Heatmap for Image 22 (Left)
Row (i)
Column (j)

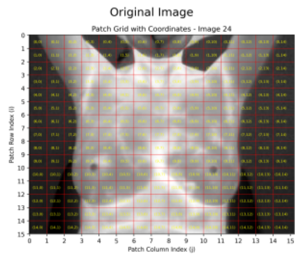

Ground Truth

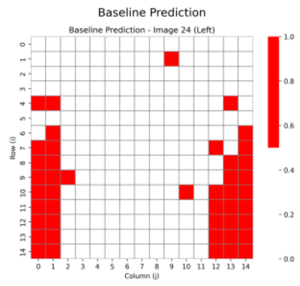

Original Image
Patch Grid with Coordinates - Image 23
Patch Row Index (i)
Patch Column Index (j)

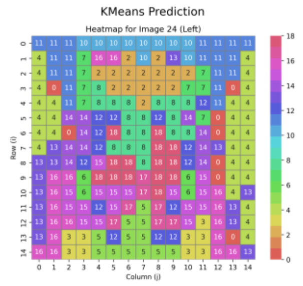

Baseline Prediction
Baseline Prediction - Image 23 (Right)
Row (i)
Column (j)

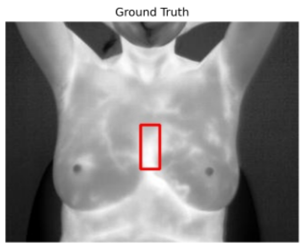

KMeans Prediction
Heatmap for Image 23 (Right)
Row (i)
Column (j)

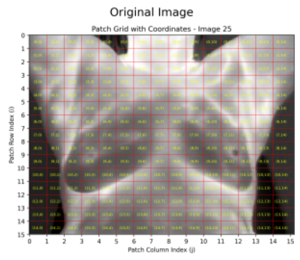

Ground Truth

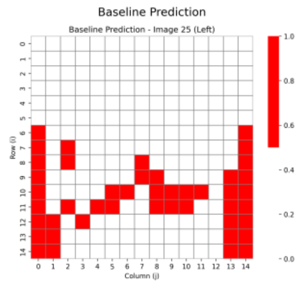

Original Image
Patch Grid with Coordinates - Image 24
Patch Row Index (i)
Patch Column Index (j)

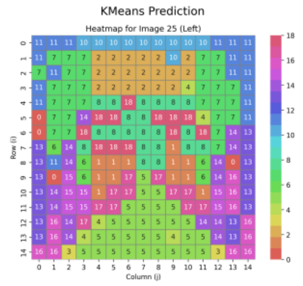

Baseline Prediction
Baseline Prediction - Image 24 (Left)
Row (i)
Column (j)

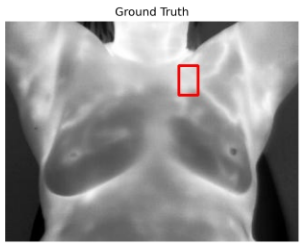

KMeans Prediction
Heatmap for Image 24 (Left)
Row (i)
Column (j)

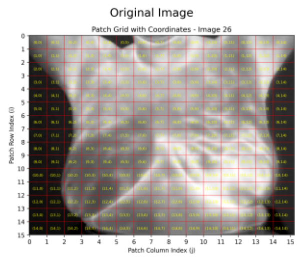

Ground Truth

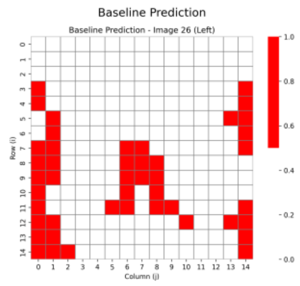

Original Image
Patch Grid with Coordinates - Image 25
Patch Row Index (i)
Patch Column Index (j)

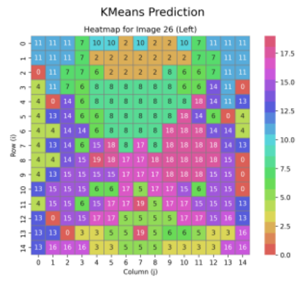

Baseline Prediction
Baseline Prediction - Image 25 (Left)
Row (i)
Column (j)

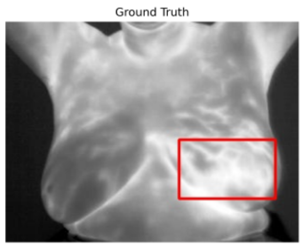

KMeans Prediction
Heatmap for Image 25 (Left)
Row (i)
Column (j)

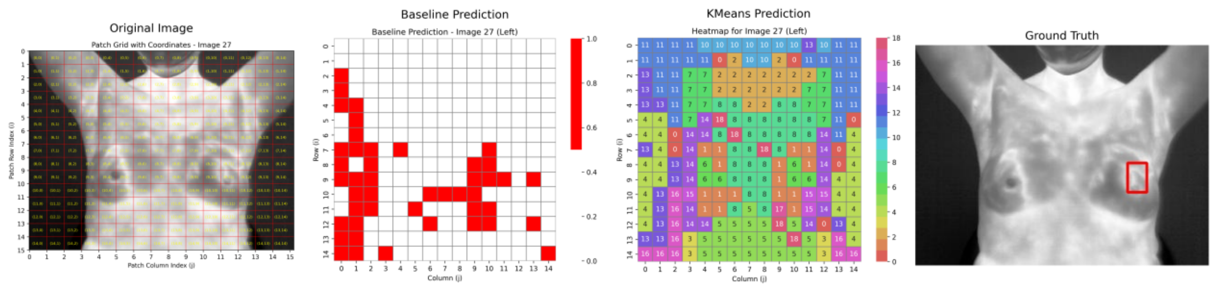

Ground Truth

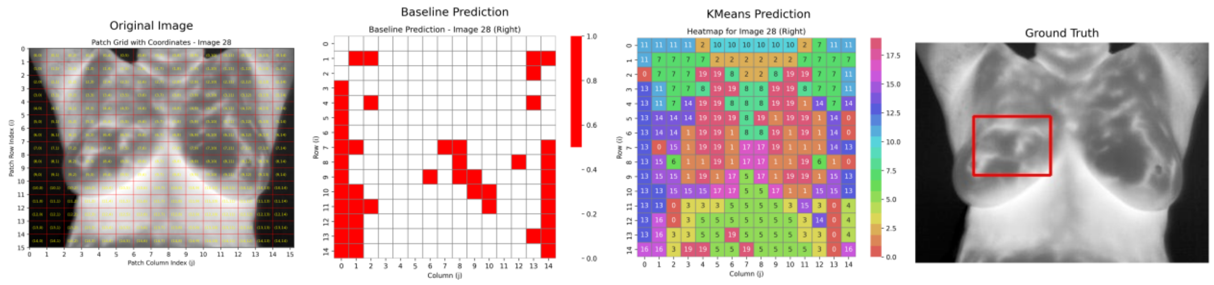

Original Image
Patch Grid with Coordinates - Image 26
Patch Row Index (i)
Patch Column Index (j)

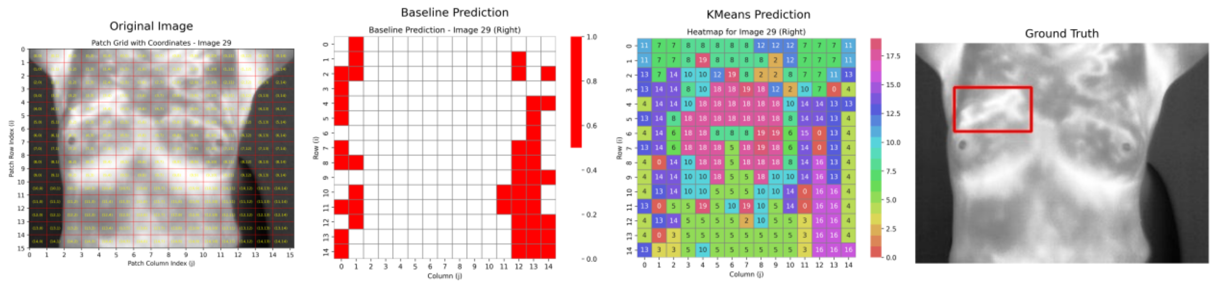

Baseline Prediction
Baseline Prediction - Image 26 (Left)
Row (i)
Column (j)

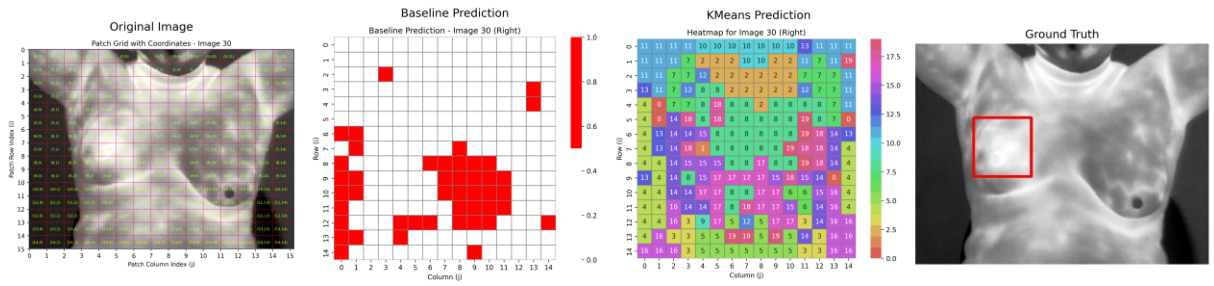

KMeans Prediction
Heatmap for Image 26 (Left)
Row (i)
Column (j)

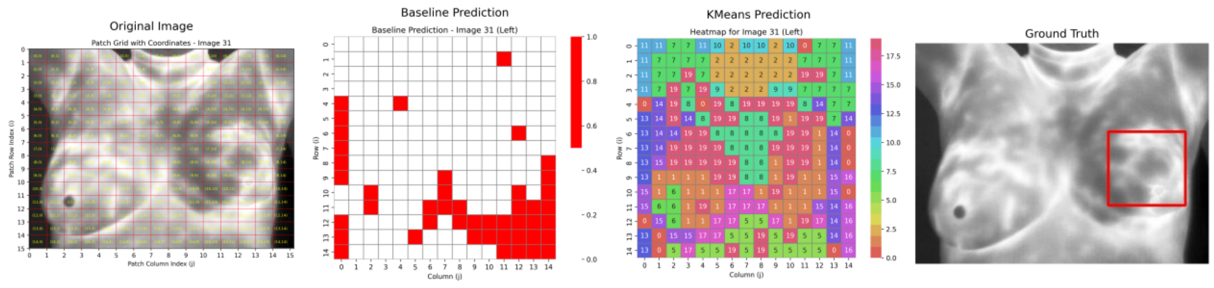

Ground Truth

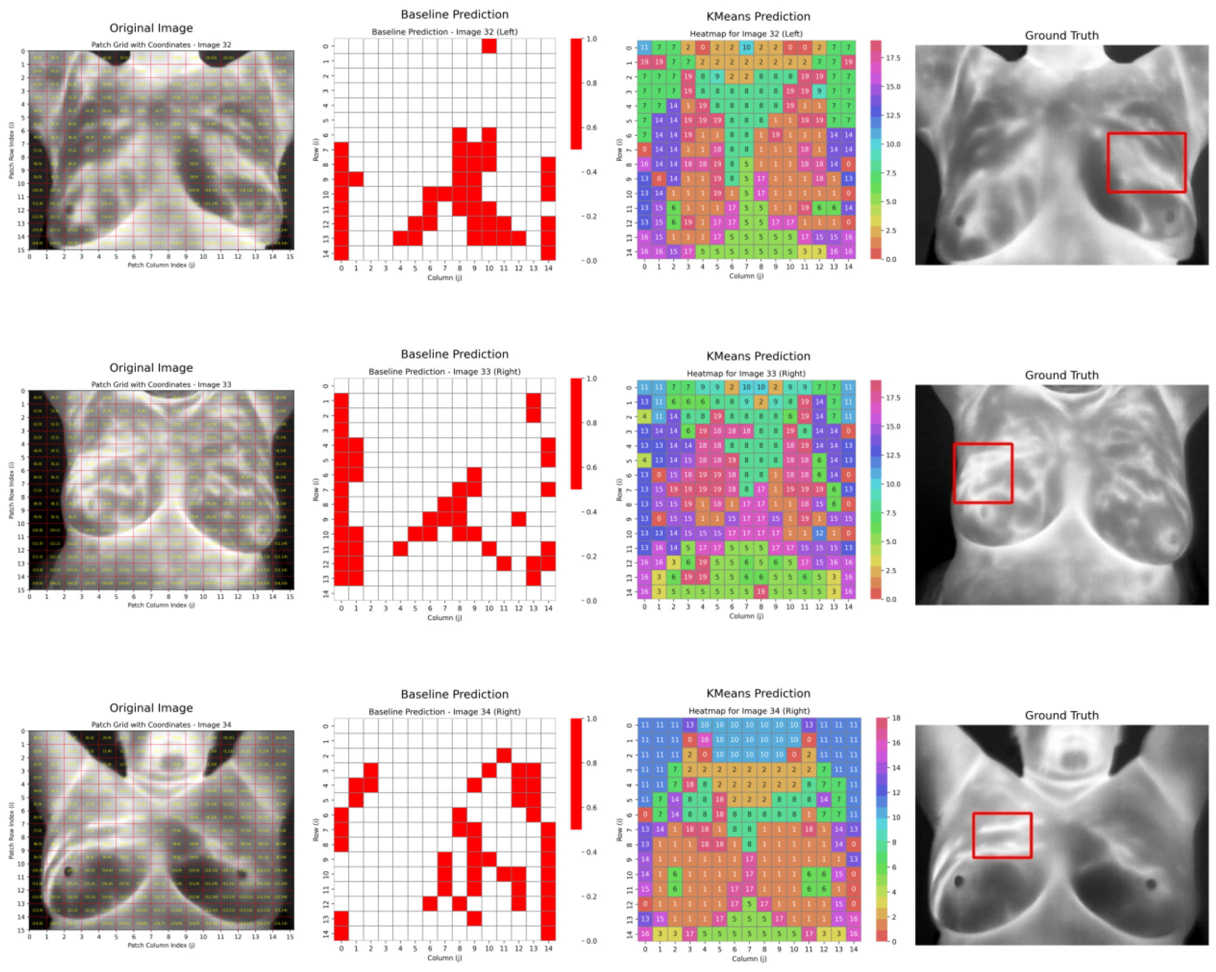

Original Image
Patch Grid with Coordinates - Image 27
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 27 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 27 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 28
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 28 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 28 (Right)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 29
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 29 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 29 (Right)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 30
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 30 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 30 (Right)
Row (i)
Column (j)
Ground Truth

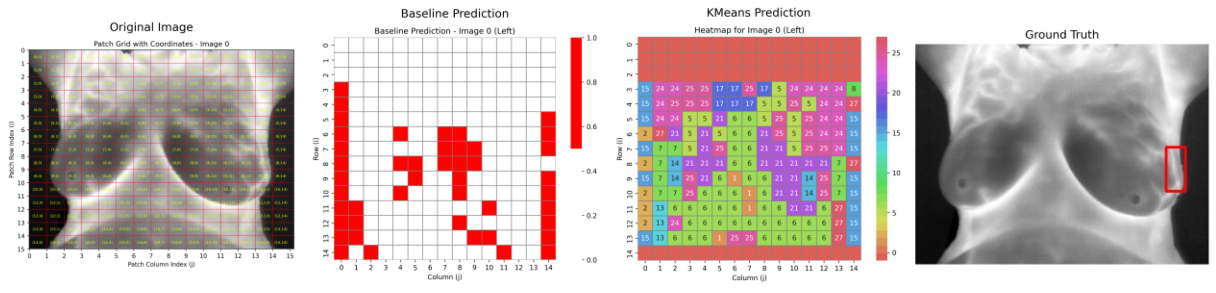

Original Image
Patch Grid with Coordinates - Image 31
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 31 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 31 (Left)
Row (i)
Column (j)
Ground Truth

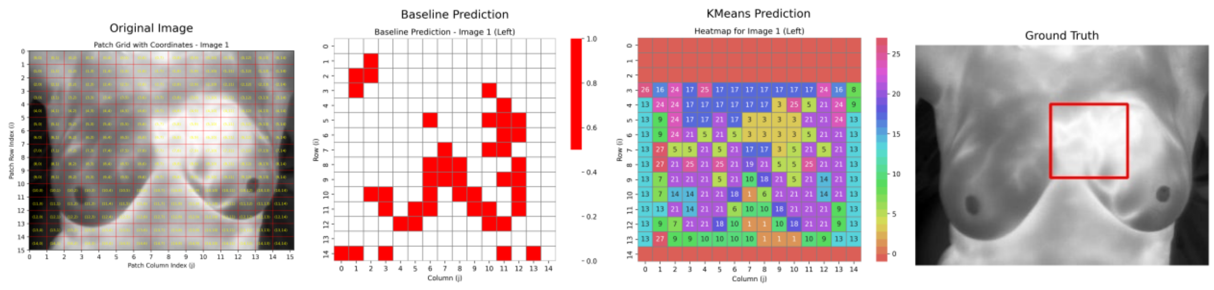

Original Image
Patch Grid with Coordinates - Image 32
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 32 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 32 (Left)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 33
Baseline Prediction
Baseline Prediction - Image 33 (Right)
KMeans Prediction
Heatmap for Image 33 (Right)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 34
Baseline Prediction
Baseline Prediction - Image 34 (Right)
KMeans Prediction
Heatmap for Image 34 (Right)
Ground Truth

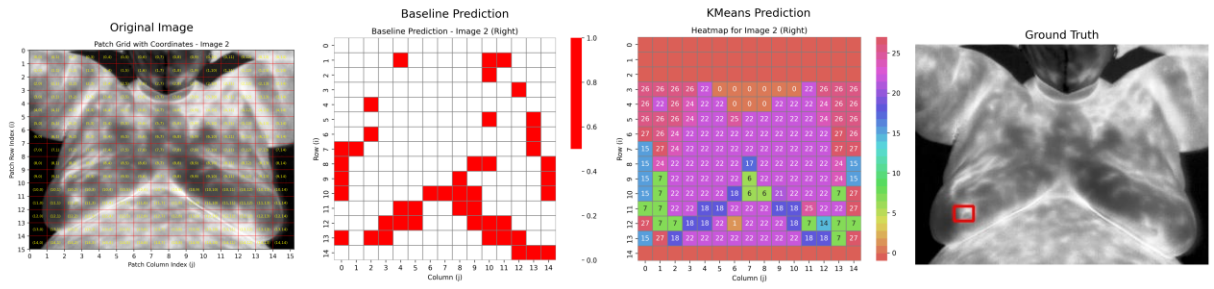
Plot 1:Comparison with 15X15 original image, baseline, k-means prediction using the experiment 5.1.1.4.2 with 20 clusters and the GT

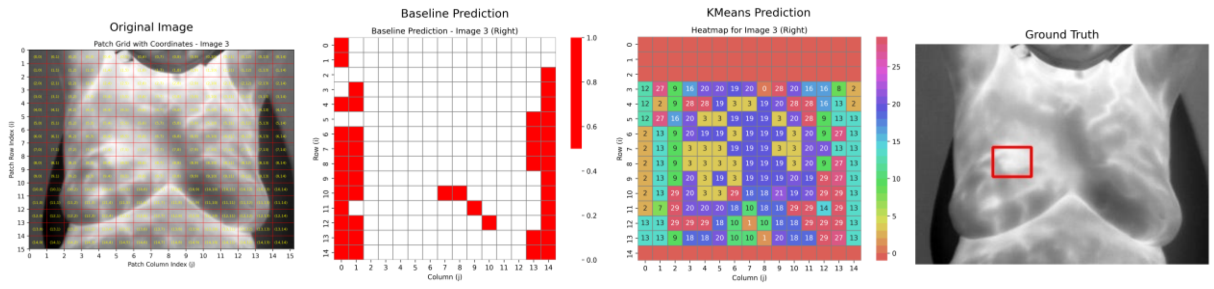
5.2.3.2 Patch selection and tumour cluster reassign using Euclidean on Raw features

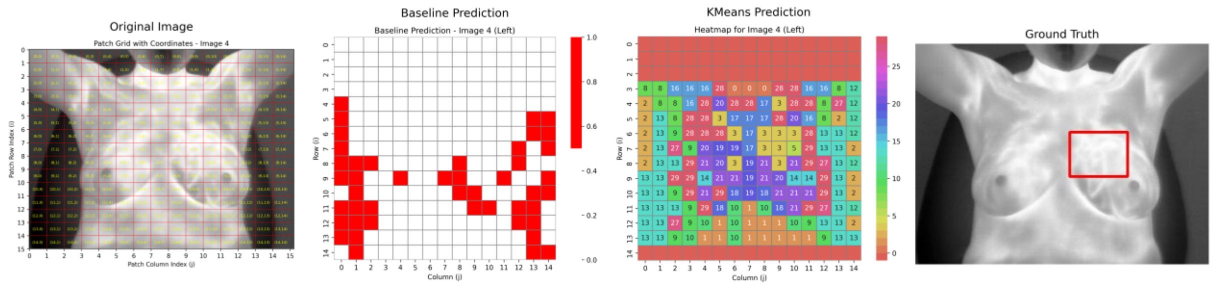
We can see the comparison with the experiment using 15X15 patches map, raw features, patches reassigning with Euclidean distance, patch selection and 30 clusters and the tumour clusters is the 3 [Plot 2].

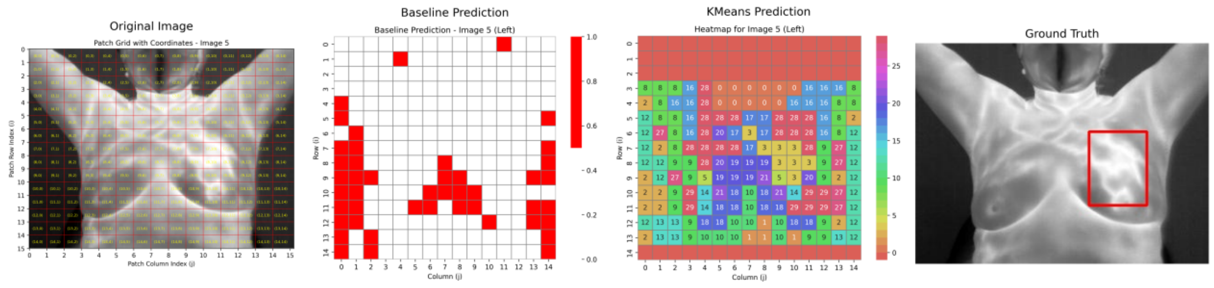


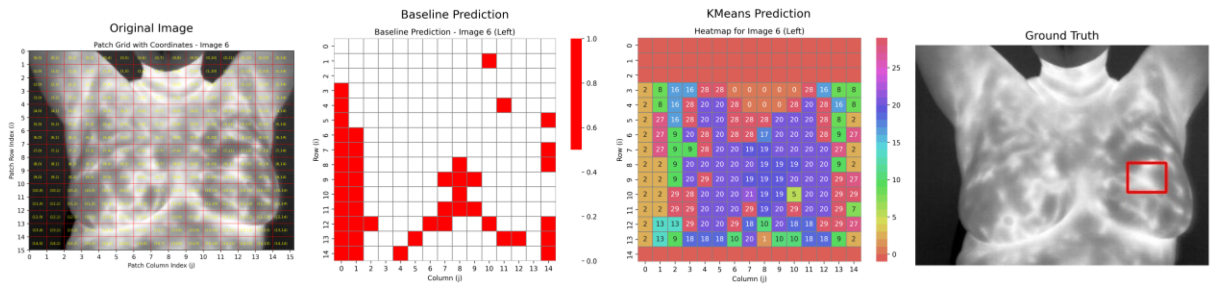

Original Image
Patch Grid with Coordinates - Image 1
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 1 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 1 (Left)
Row (i)
Column (j)
Ground Truth

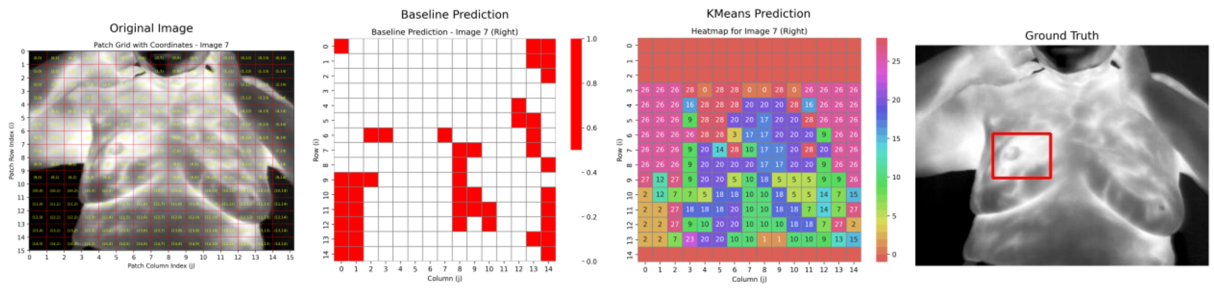

Original Image
Patch Grid with Coordinates - Image 2
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 2 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 2 (Right)
Row (i)
Column (j)
Ground Truth

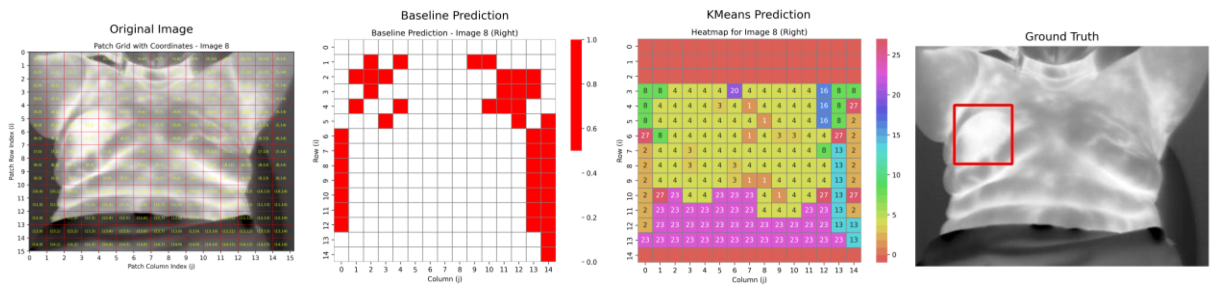

Original Image
Patch Grid with Coordinates - Image 3
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 3 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 3 (Right)
Row (i)
Column (j)
Ground Truth

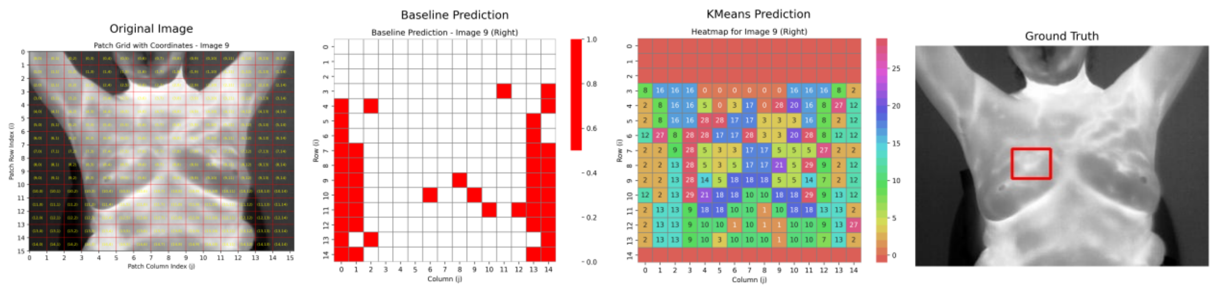

Original Image
Patch Grid with Coordinates - Image 4
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 4 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 4 (Left)
Row (i)
Column (j)
Ground Truth

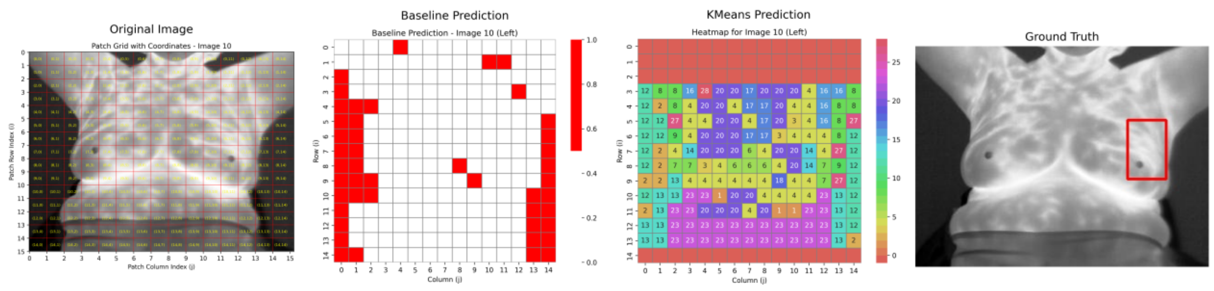

Original Image
Patch Grid with Coordinates - Image 5
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 5 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 5 (Left)
Row (i)
Column (j)
Ground Truth

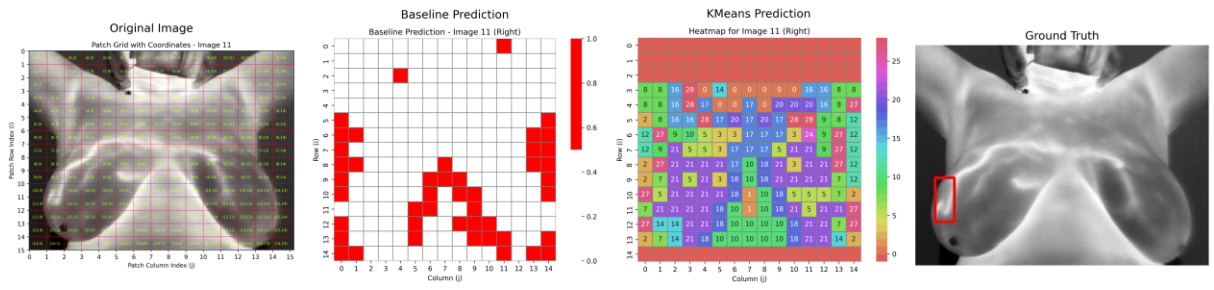

Original Image
Patch Grid with Coordinates - Image 6
Baseline Prediction
Baseline Prediction - Image 6 (Left)
KMeans Prediction
Heatmap for Image 6 (Left)
Ground Truth

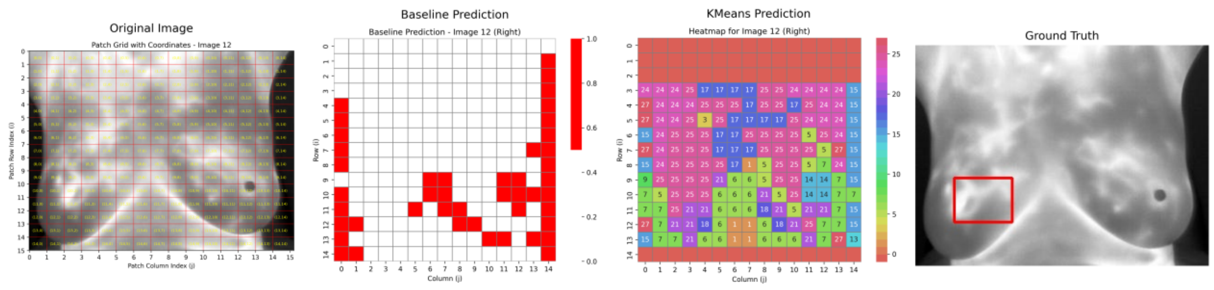

Original Image
Patch Grid with Coordinates - Image 7
Baseline Prediction
Baseline Prediction - Image 7 (Right)
KMeans Prediction
Heatmap for Image 7 (Right)
Ground Truth

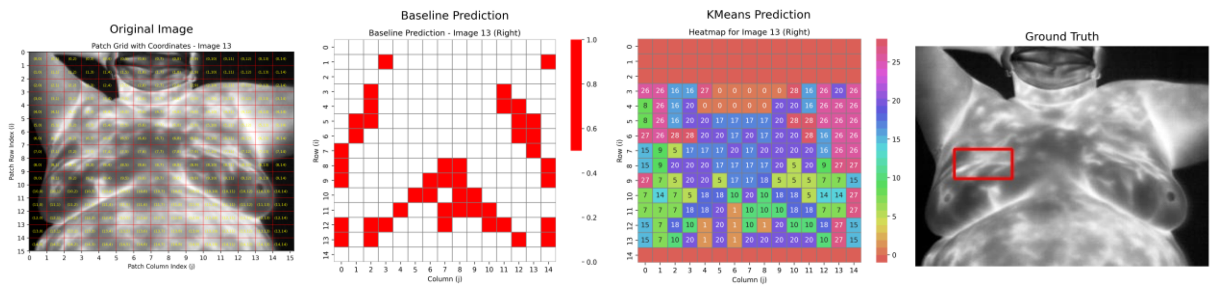

Original Image
Patch Grid with Coordinates - Image 8
Baseline Prediction
Baseline Prediction - Image 8 (Right)
KMeans Prediction
Heatmap for Image 8 (Right)
Ground Truth

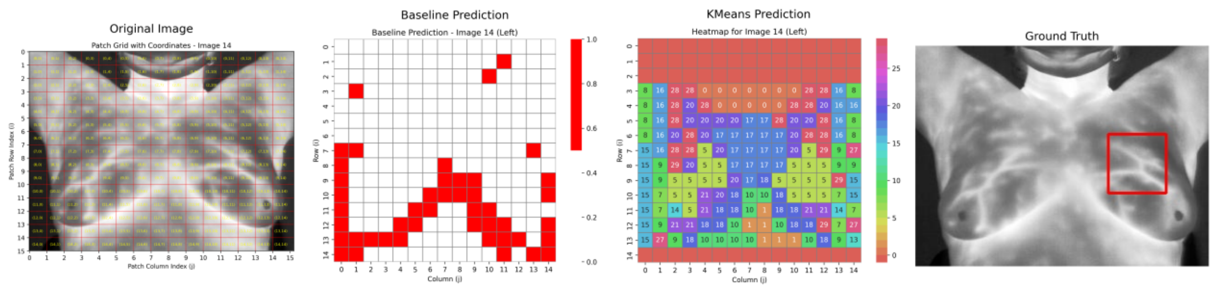

Original Image
Patch Grid with Coordinates - Image 9
Baseline Prediction
Baseline Prediction - Image 9 (Right)
KMeans Prediction
Heatmap for Image 9 (Right)
Ground Truth

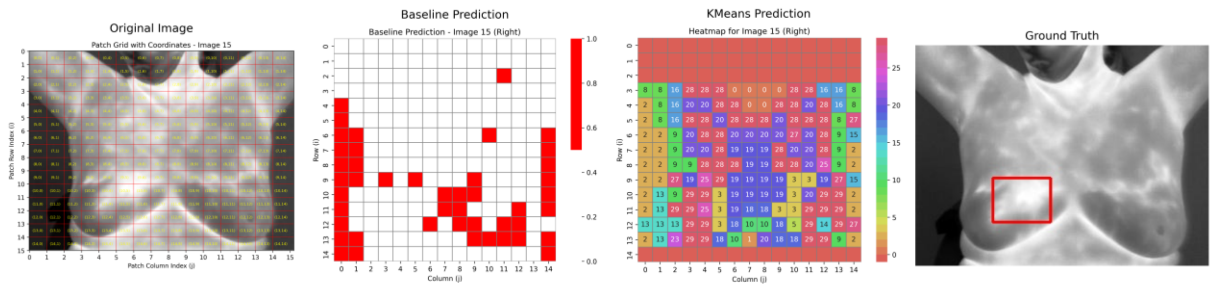

Original Image
Patch Grid with Coordinates - Image 10
Baseline Prediction
Baseline Prediction - Image 10 (Left)
KMeans Prediction
Heatmap for Image 10 (Left)
Ground Truth

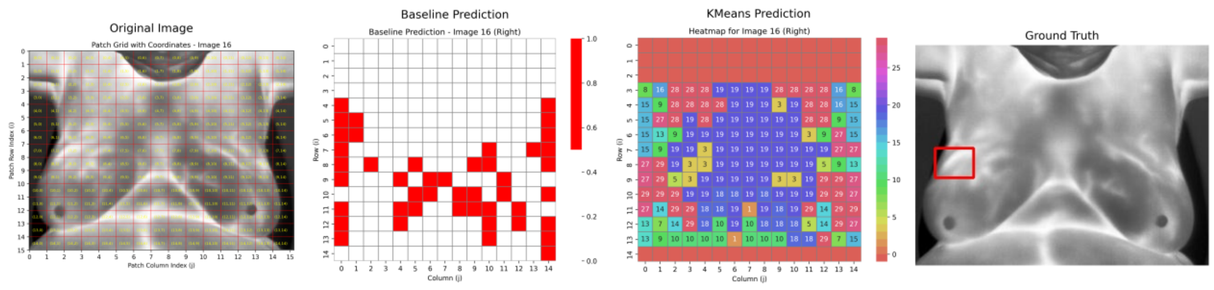

Original Image
Patch Grid with Coordinates - Image 11
Baseline Prediction
Baseline Prediction - Image 11 (Right)
KMeans Prediction
Heatmap for Image 11 (Right)
Ground Truth

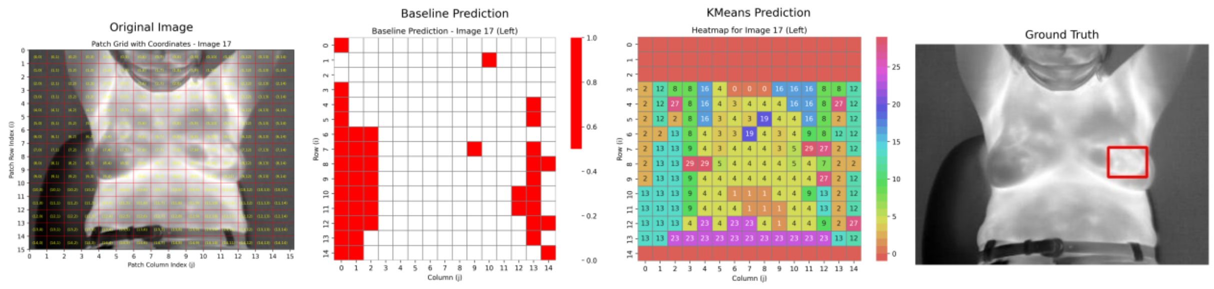

Original Image
Patch Grid with Coordinates - Image 12
Baseline Prediction
Baseline Prediction - Image 12 (Right)
KMeans Prediction
Heatmap for Image 12 (Right)
Ground Truth

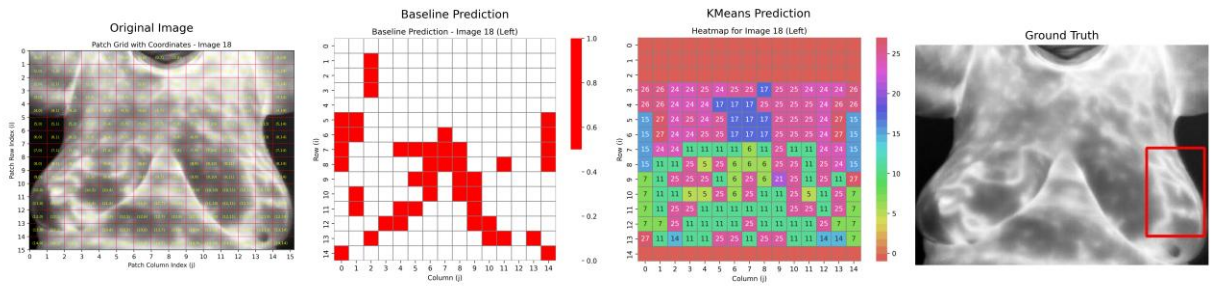

Original Image
Patch Grid with Coordinates - Image 13
Baseline Prediction
Baseline Prediction - Image 13 (Right)
KMeans Prediction
Heatmap for Image 13 (Right)
Ground Truth

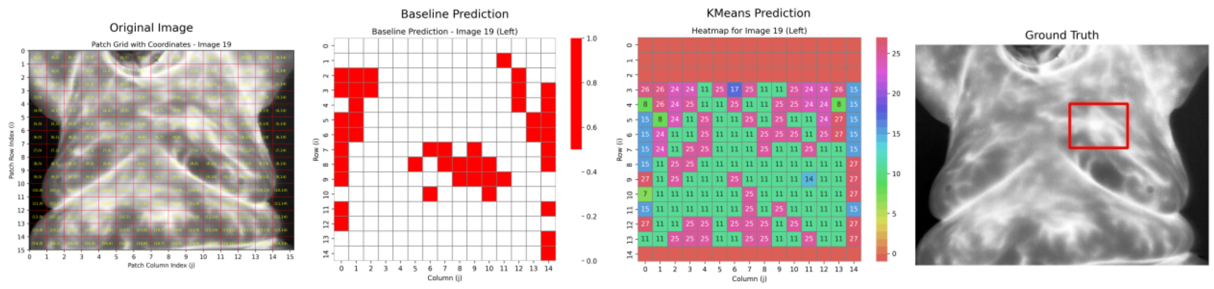

Original Image
Patch Grid with Coordinates - Image 14
Baseline Prediction
Baseline Prediction - Image 14 (Left)
KMeans Prediction
Heatmap for Image 14 (Left)
Ground Truth

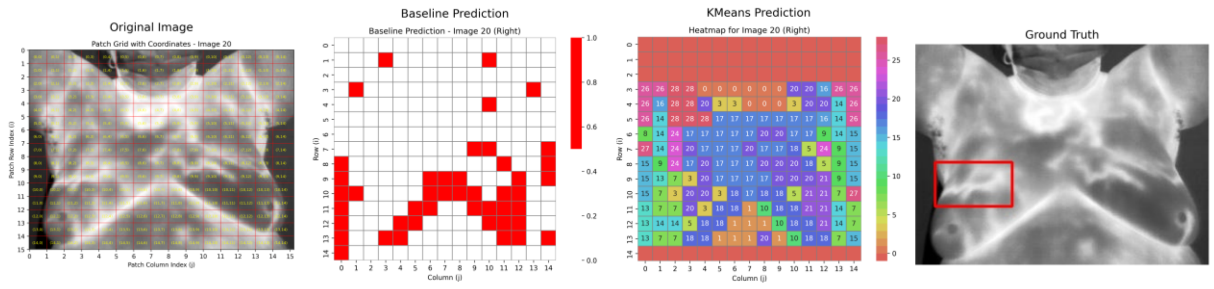

Original Image
Patch Grid with Coordinates - Image 15
Baseline Prediction
Baseline Prediction - Image 15 (Right)
KMeans Prediction
Heatmap for Image 15 (Right)
Ground Truth

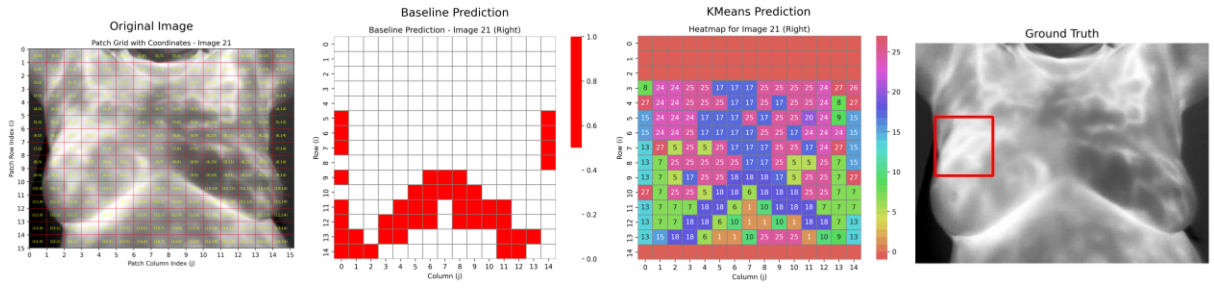

Original Image
Patch Grid with Coordinates - Image 16
Baseline Prediction
Baseline Prediction - Image 16 (Right)
KMeans Prediction
Heatmap for Image 16 (Right)
Ground Truth

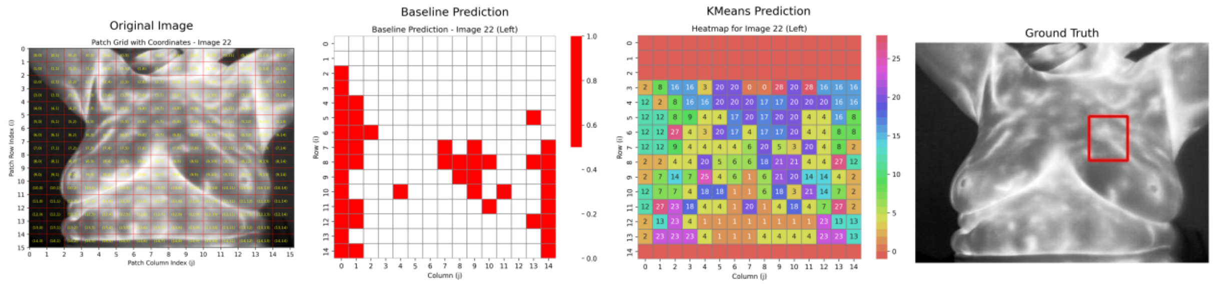

Original Image
Patch Grid with Coordinates - Image 17
Baseline Prediction
Baseline Prediction - Image 17 (Left)
KMeans Prediction
Heatmap for Image 17 (Left)
Ground Truth

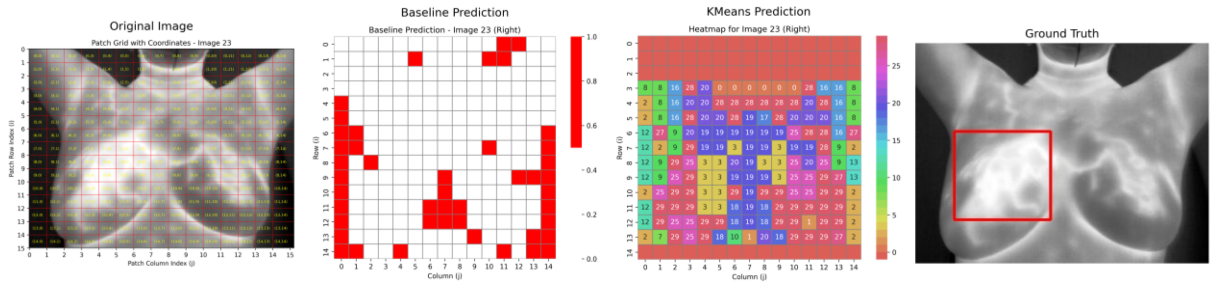

Original Image
Patch Grid with Coordinates - Image 18
Baseline Prediction
Baseline Prediction - Image 18 (Left)
KMeans Prediction
Heatmap for Image 18 (Left)
Ground Truth

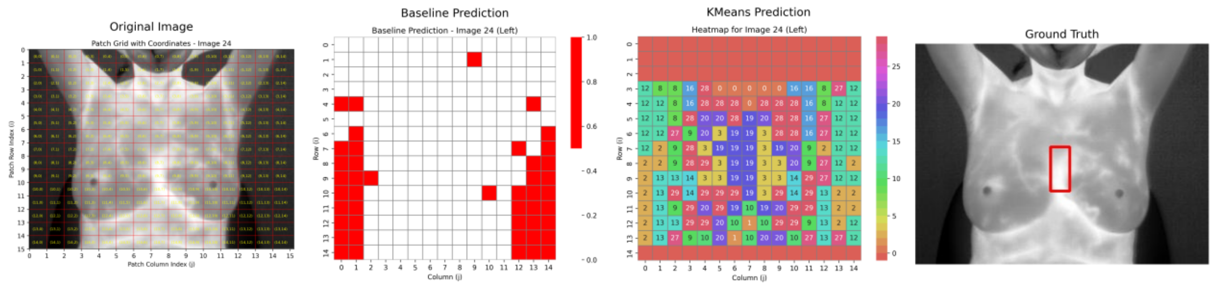

Original Image
Patch Grid with Coordinates - Image 19
Baseline Prediction
Baseline Prediction - Image 19 (Left)
KMeans Prediction
Heatmap for Image 19 (Left)
Ground Truth

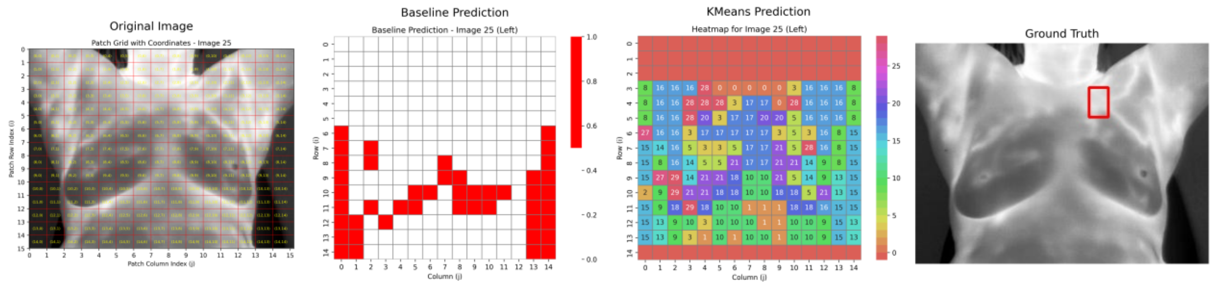

Original Image
Patch Grid with Coordinates - Image 20
Baseline Prediction
Baseline Prediction - Image 20 (Right)
KMeans Prediction
Heatmap for Image 20 (Right)
Ground Truth

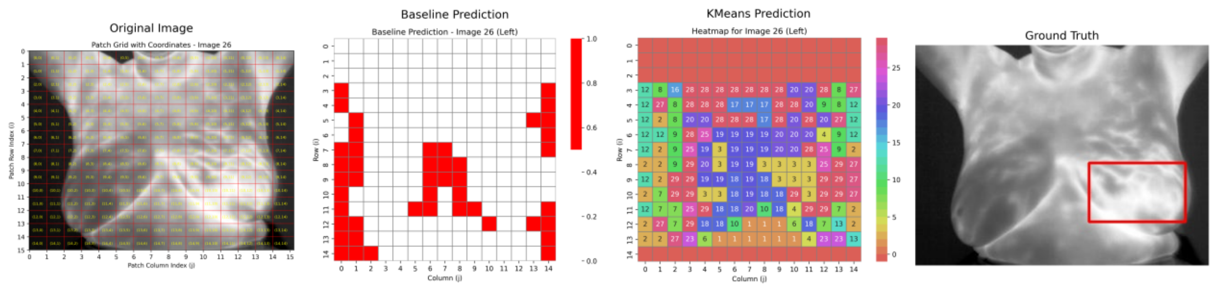

Original Image
Patch Grid with Coordinates - Image 21
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 21 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 21 (Right)
Row (i)
Column (j)
Ground Truth

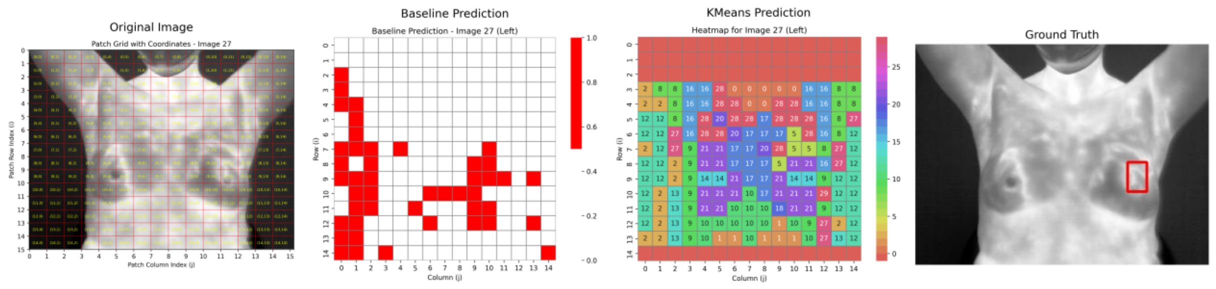

Original Image
Patch Grid with Coordinates - Image 22
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 22 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 22 (Left)
Row (i)
Column (j)
Ground Truth

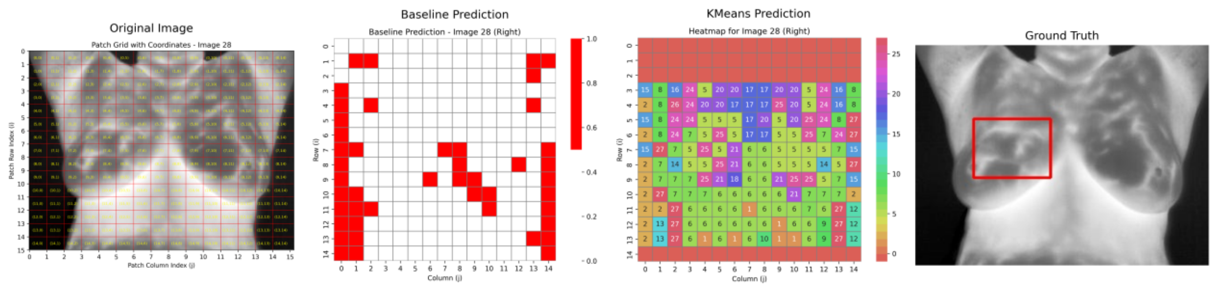

Original Image
Patch Grid with Coordinates - Image 23
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 23 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 23 (Right)
Row (i)
Column (j)
Ground Truth

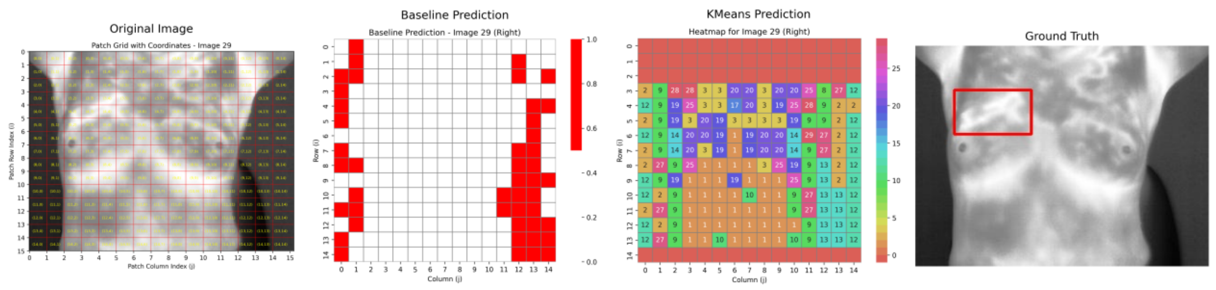

Original Image
Patch Grid with Coordinates - Image 24
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 24 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 24 (Left)
Row (i)
Column (j)
Ground Truth

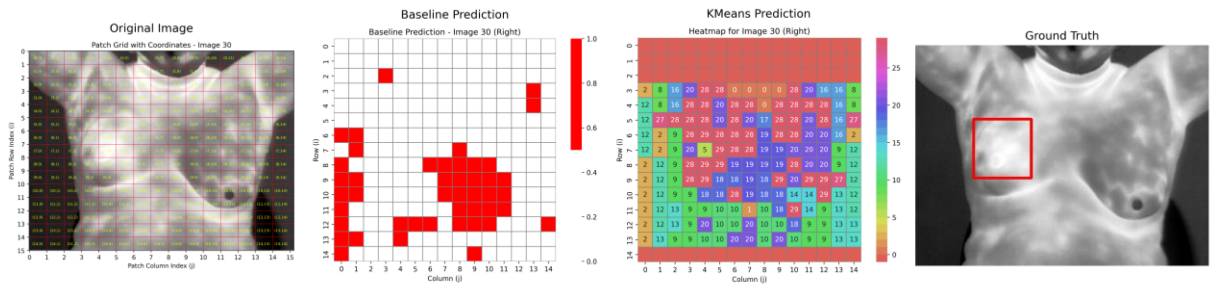

Original Image
Patch Grid with Coordinates - Image 25
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 25 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 25 (Left)
Row (i)
Column (j)
Ground Truth

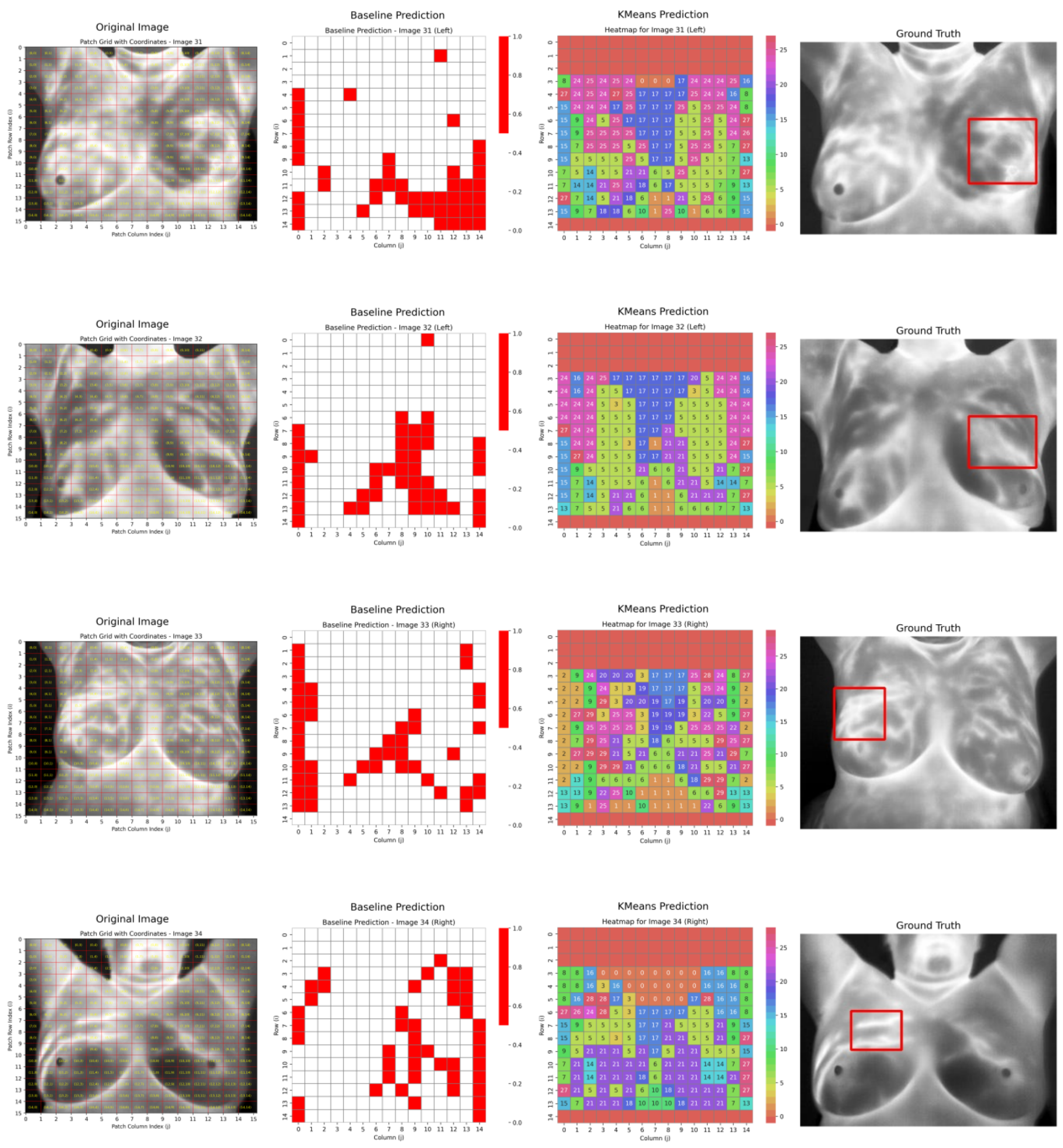

Original Image
Patch Grid with Coordinates - Image 26
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 26 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 26 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 27
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 27 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 27 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 28
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 28 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 28 (Right)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 29
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 29 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 29 (Right)
Row (i)
Column (j)
Ground Truth

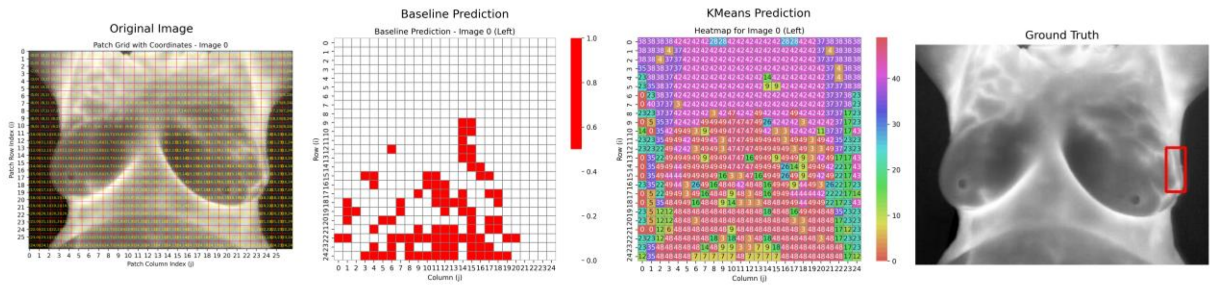

Original Image
Patch Grid with Coordinates - Image 30
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 30 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 30 (Right)
Row (i)
Column (j)
Ground Truth

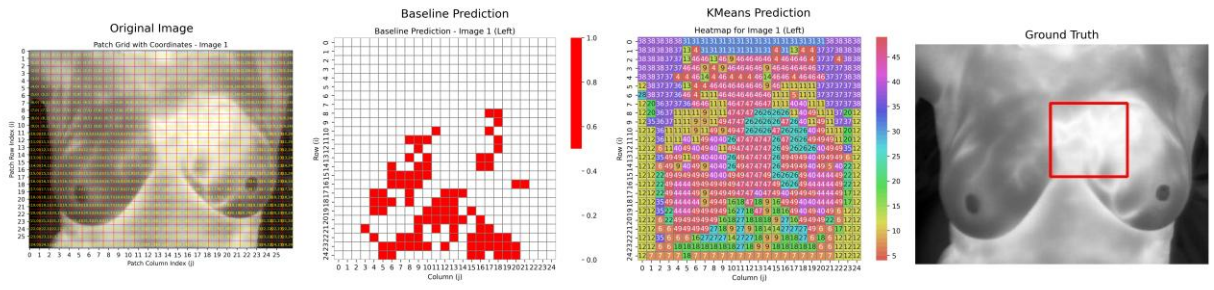

Original Image
Patch Grid with Coordinates - Image 31
Baseline Prediction
Baseline Prediction - Image 31 (Left)
KMeans Prediction
Heatmap for Image 31 (Left)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 32
Baseline Prediction
Baseline Prediction - Image 32 (Left)
KMeans Prediction
Heatmap for Image 32 (Left)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 33
Baseline Prediction
Baseline Prediction - Image 33 (Right)
KMeans Prediction
Heatmap for Image 33 (Right)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 34
Baseline Prediction
Baseline Prediction - Image 34 (Right)
KMeans Prediction
Heatmap for Image 34 (Right)
Ground Truth
Patch Row Index (i)
Patch Column Index (j)
Row (i)
Column (j)

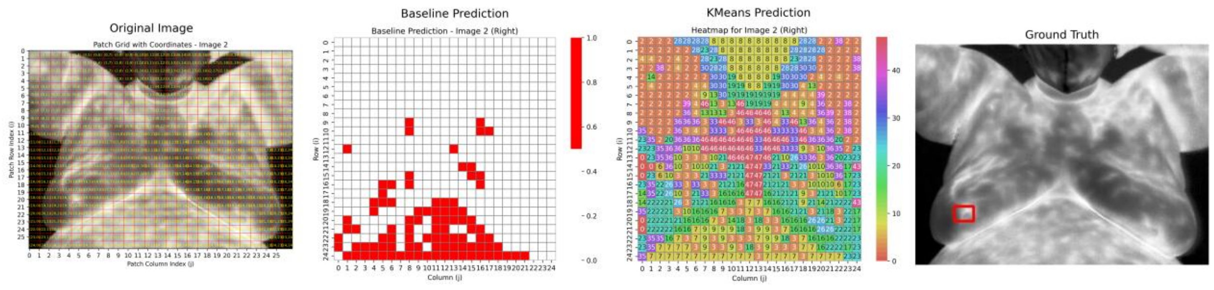
Plot 2:Comparison with 15X15 original image, baseline, k-means prediction using the experiment 5.1.1.7 with raw features and the GT

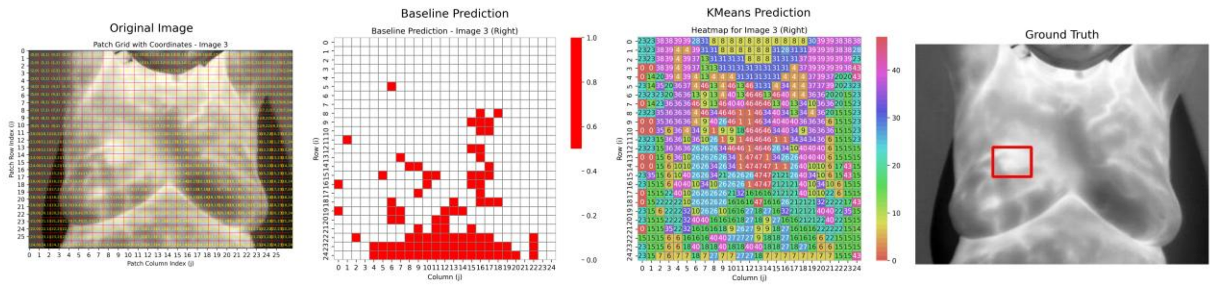
5.2.3.2 Patch selection and tumour cluster reassign using Euclidean on Raw features

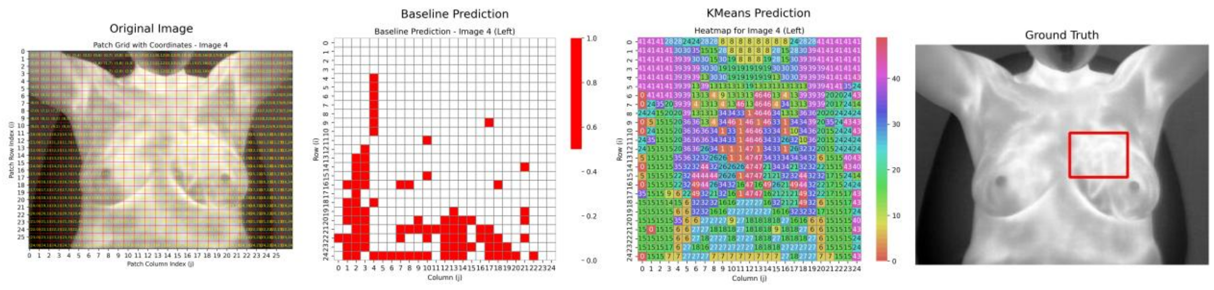
Lastly can see the comparison with the experiment using 25X15 patches map, raw features, patches reassigning with Euclidean distance and 50 clusters and the tumour clusters is the 26 [Plot 3].

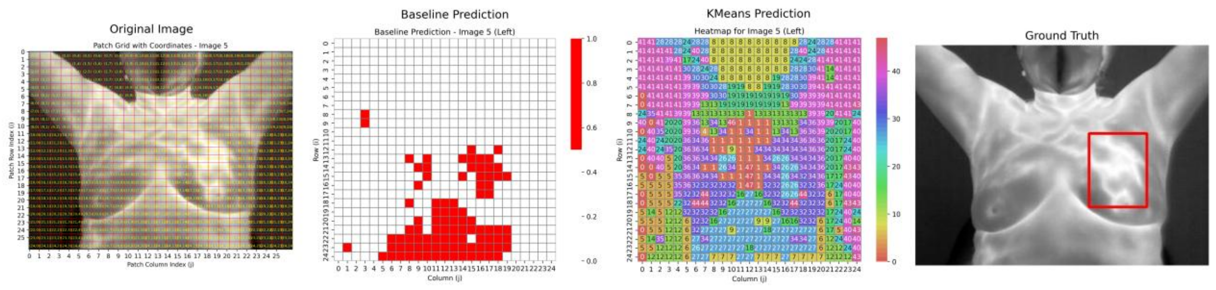

Original Image
Patch Grid with Coordinates - Image 0
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 0 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 0 (Left)
Row (i)
Column (j)
Ground Truth

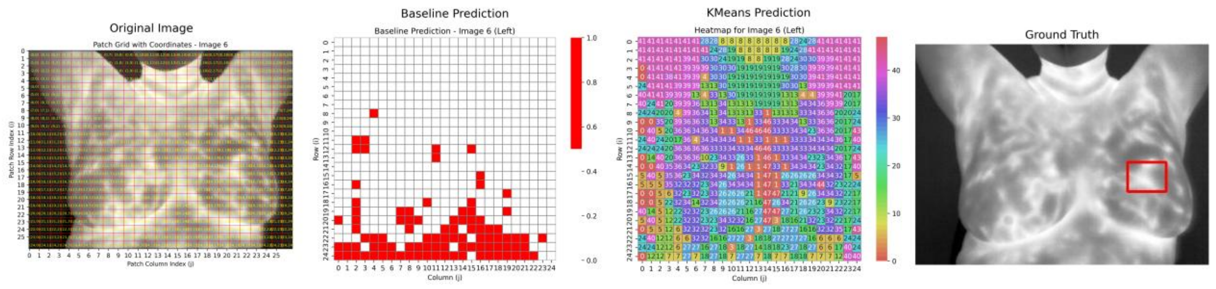

Original Image
Patch Grid with Coordinates - Image 1
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 1 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 1 (Left)
Row (i)
Column (j)
Ground Truth

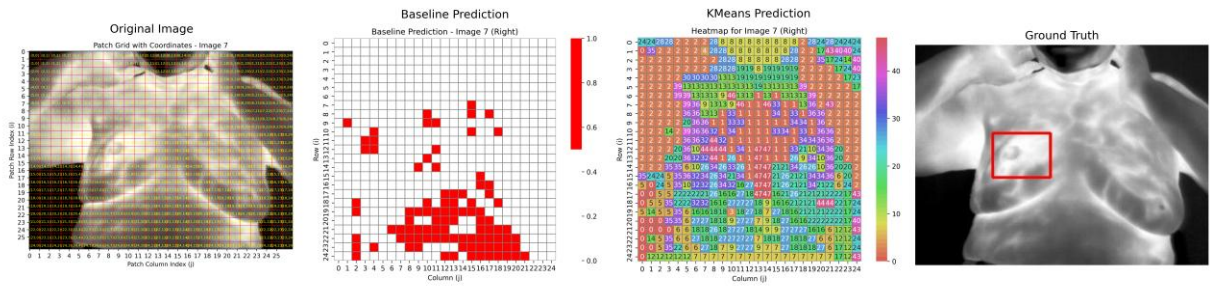

Original Image
Patch Grid with Coordinates - Image 2
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 2 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 2 (Right)
Row (i)
Column (j)
Ground Truth

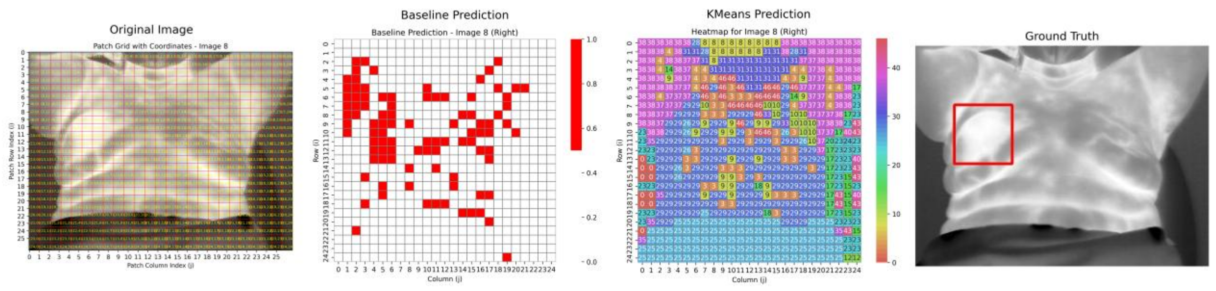

Original Image
Patch Grid with Coordinates - Image 3
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 3 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 3 (Right)
Row (i)
Column (j)
Ground Truth

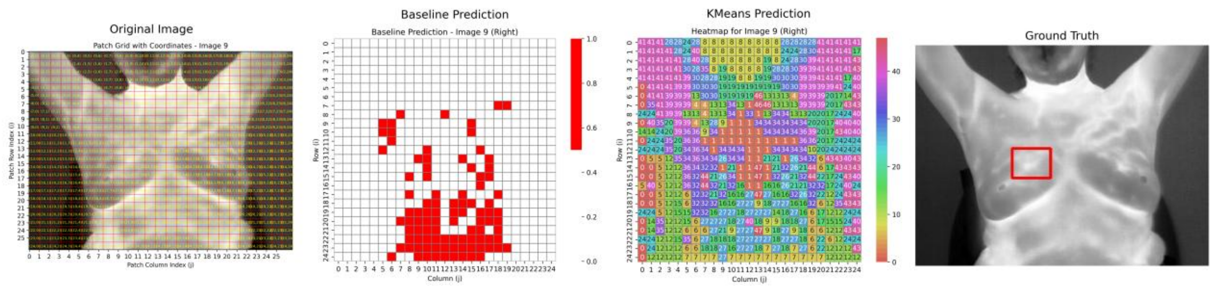

Original Image
Patch Grid with Coordinates - Image 4
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 4 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 4 (Left)
Row (i)
Column (j)
Ground Truth

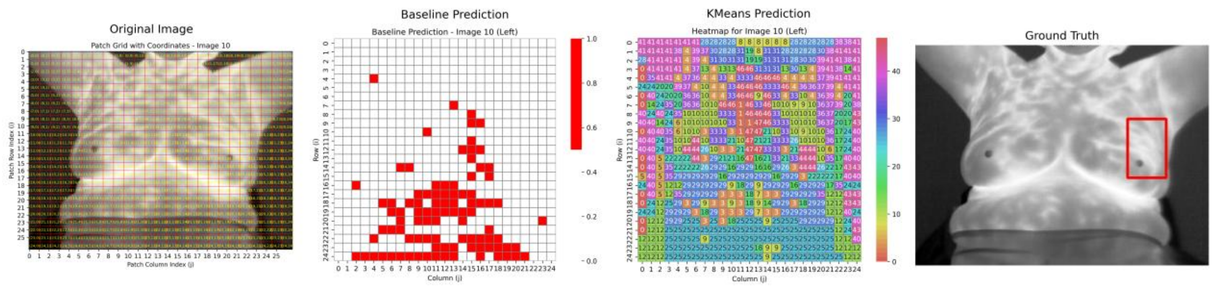

Original Image
Patch Grid with Coordinates - Image 5
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 5 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 5 (Left)
Row (i)
Column (j)
Ground Truth

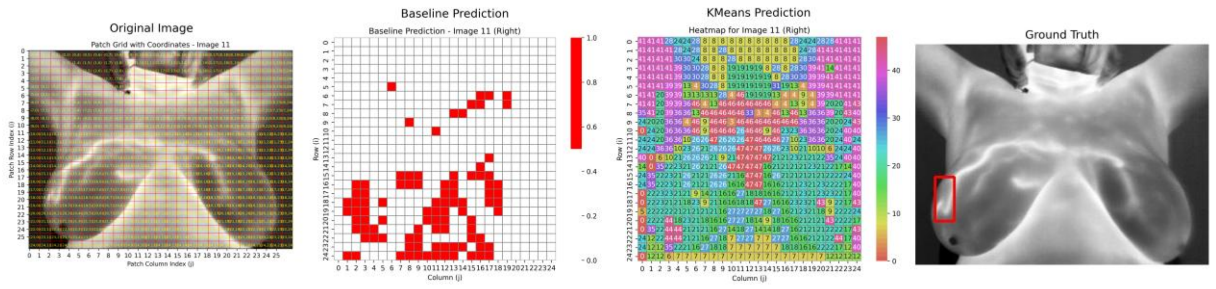

Original Image
Patch Grid with Coordinates - Image 6
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 6 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 6 (Left)
Row (i)
Column (j)
Ground Truth

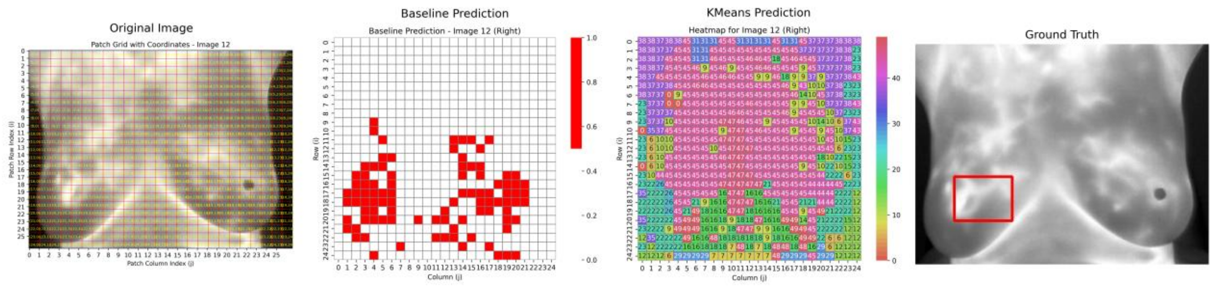

Original Image
Patch Grid with Coordinates - Image 7
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 7 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 7 (Right)
Row (i)
Column (j)
Ground Truth

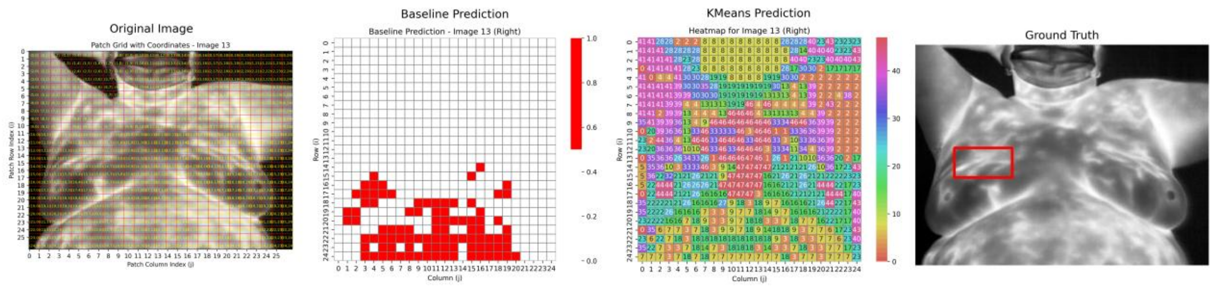

Original Image
Patch Grid with Coordinates - Image 8
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 8 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 8 (Right)
Row (i)
Column (j)
Ground Truth

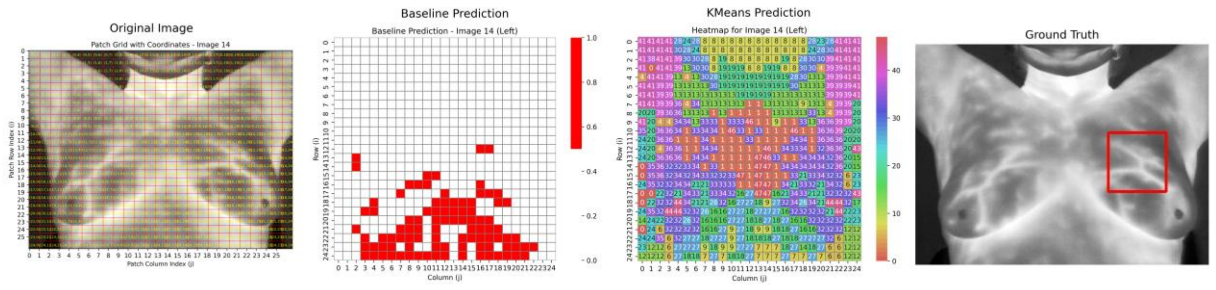

Original Image
Patch Grid with Coordinates - Image 9
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 9 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 9 (Right)
Row (i)
Column (j)
Ground Truth

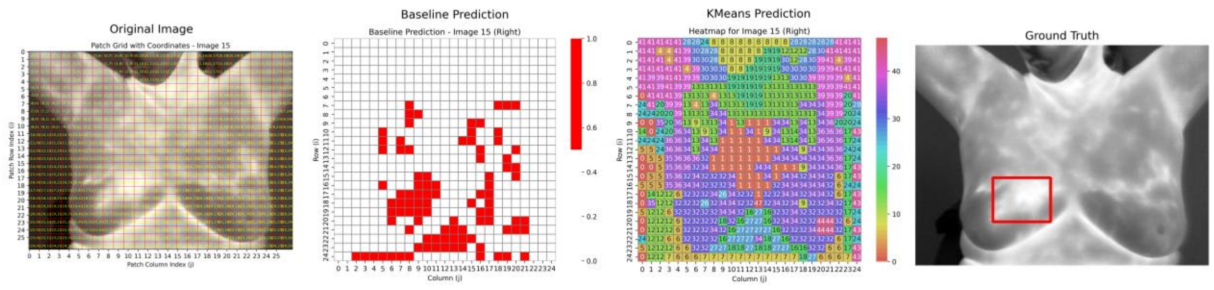

Original Image
Patch Grid with Coordinates - Image 10
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 10 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 10 (Left)
Row (i)
Column (j)
Ground Truth

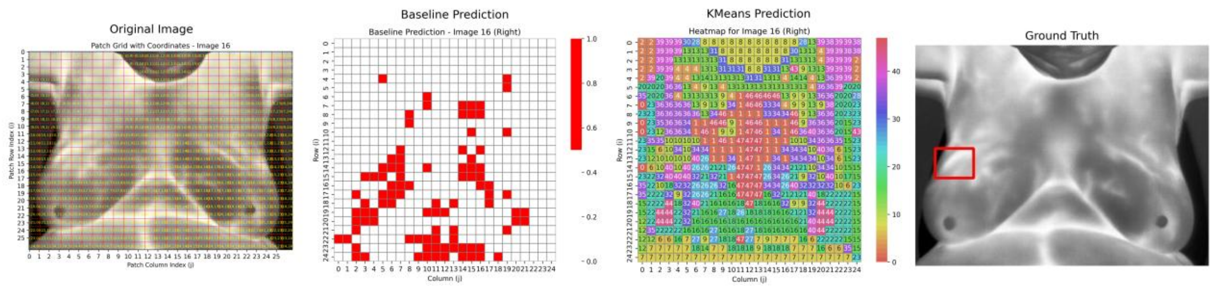

Original Image
Patch Grid with Coordinates - Image 11
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 11 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 11 (Right)
Row (i)
Column (j)
Ground Truth

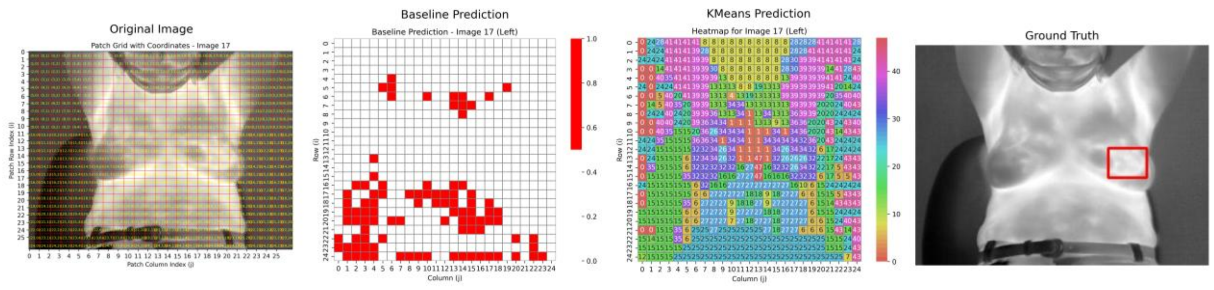

Original Image
Patch Grid with Coordinates - Image 12
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 12 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 12 (Right)
Row (i)
Column (j)
Ground Truth

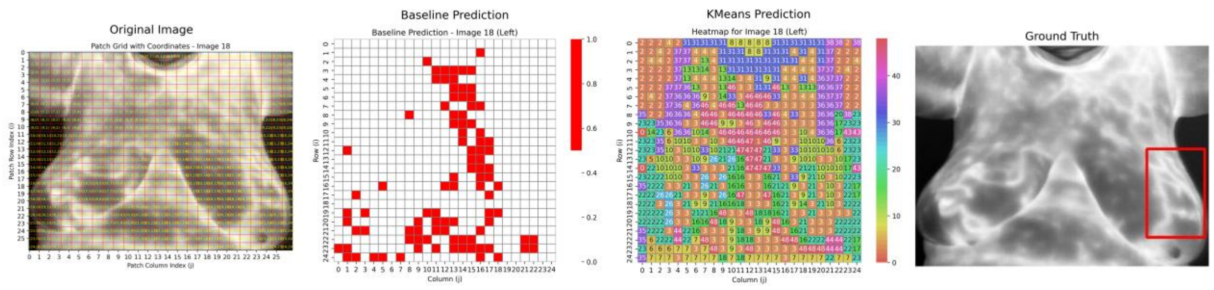

Original Image
Patch Grid with Coordinates - Image 13
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 13 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 13 (Right)
Row (i)
Column (j)
Ground Truth

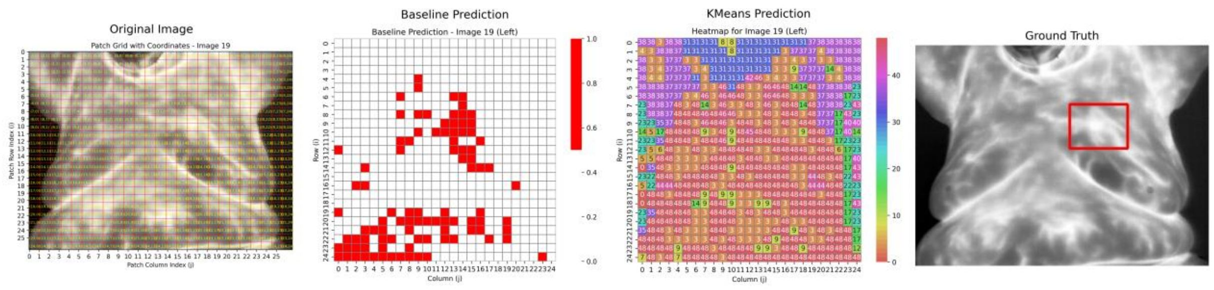

Original Image
Patch Grid with Coordinates - Image 14
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 14 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 14 (Left)
Row (i)
Column (j)
Ground Truth

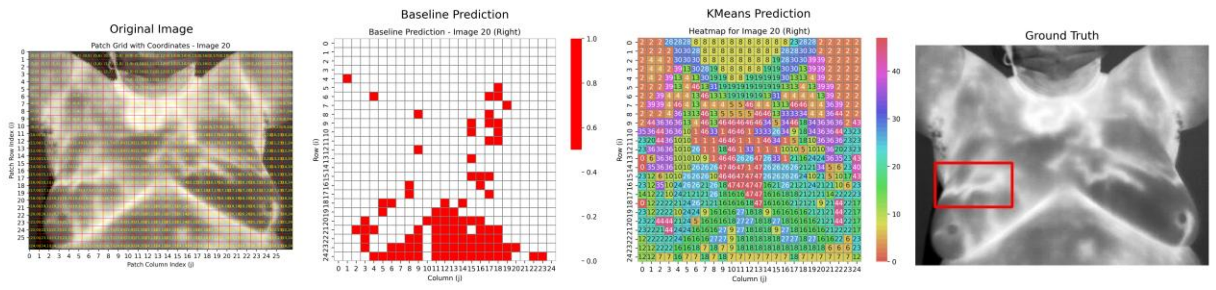

Original Image
Patch Grid with Coordinates - Image 15
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 15 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 15 (Right)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 16
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 16 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 16 (Right)
Row (i)
Column (j)
Ground Truth

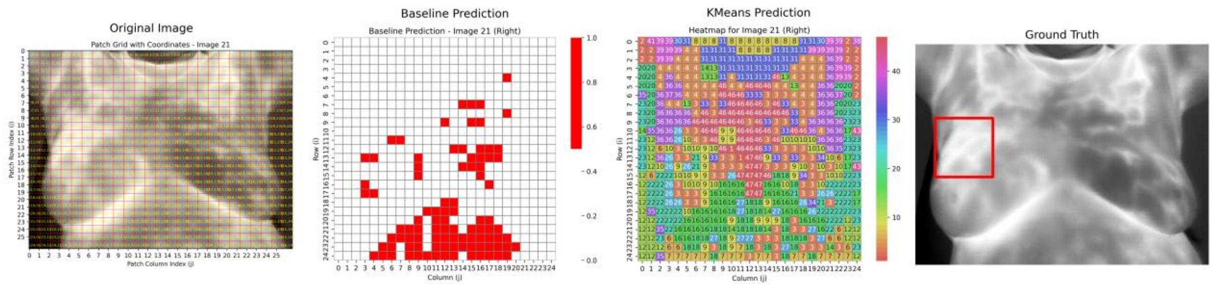

Original Image
Patch Grid with Coordinates - Image 17
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 17 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 17 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 18
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 18 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 18 (Left)
Row (i)
Column (j)
Ground Truth

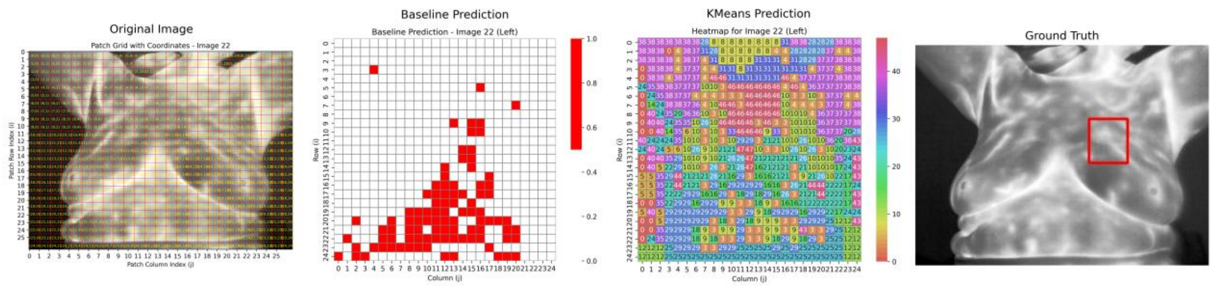

Original Image
Patch Grid with Coordinates - Image 19
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 19 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 19 (Left)
Row (i)
Column (j)
Ground Truth



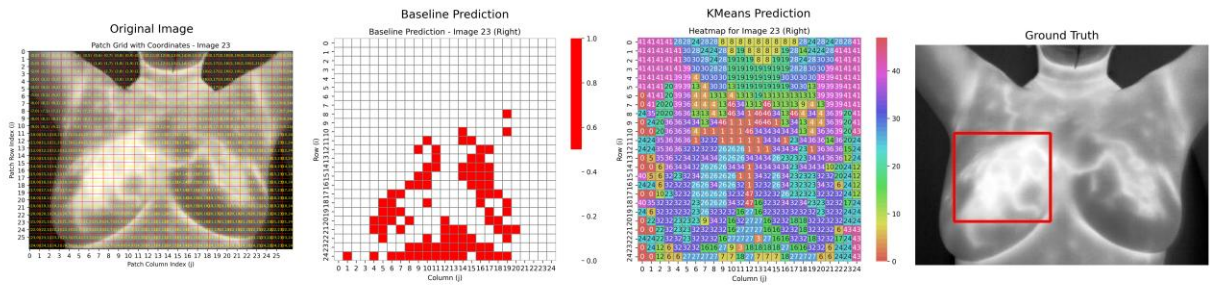
Original Image
Patch Grid with Coordinates - Image 20
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 20 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 20 (Right)
Row (i)
Column (j)
Ground Truth



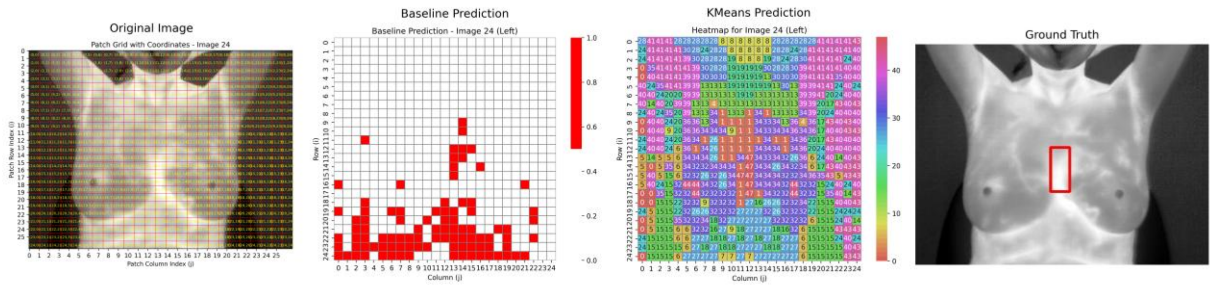
Original Image
Patch Grid with Coordinates - Image 21
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 21 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 21 (Right)
Row (i)
Column (j)
Ground Truth



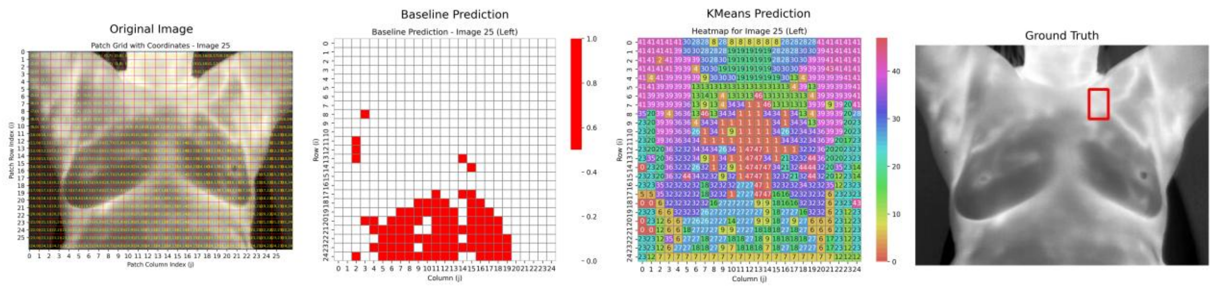
Original Image
Patch Grid with Coordinates - Image 22
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 22 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 22 (Left)
Row (i)
Column (j)
Ground Truth

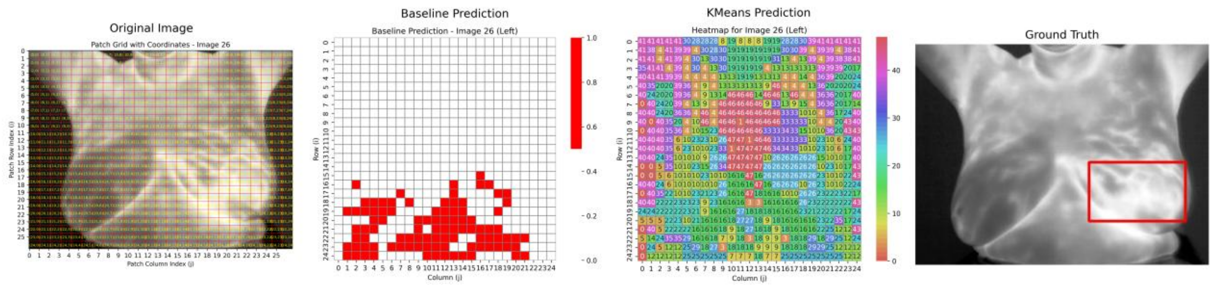


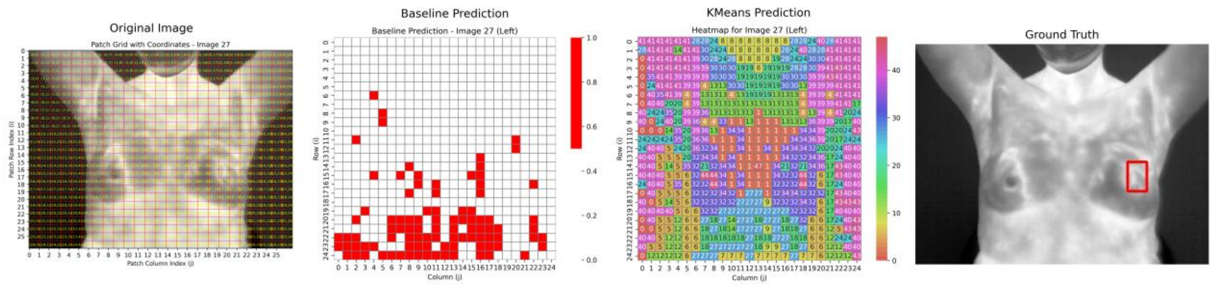
Original Image
Patch Grid with Coordinates - Image 23
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 23 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 23 (Right)
Row (i)
Column (j)
Ground Truth

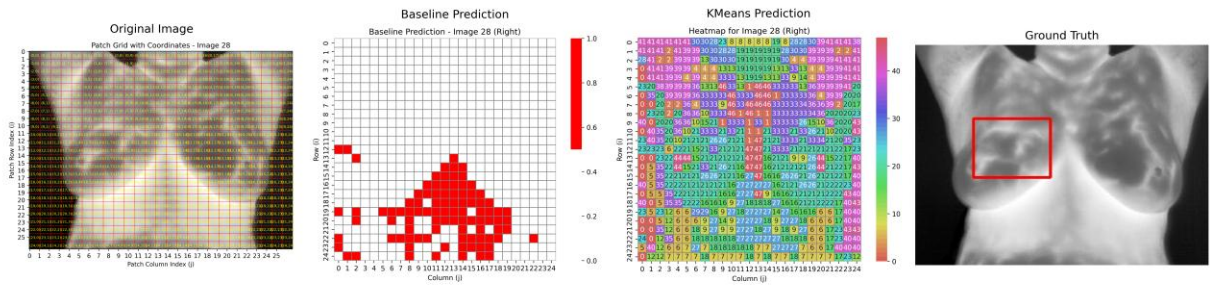


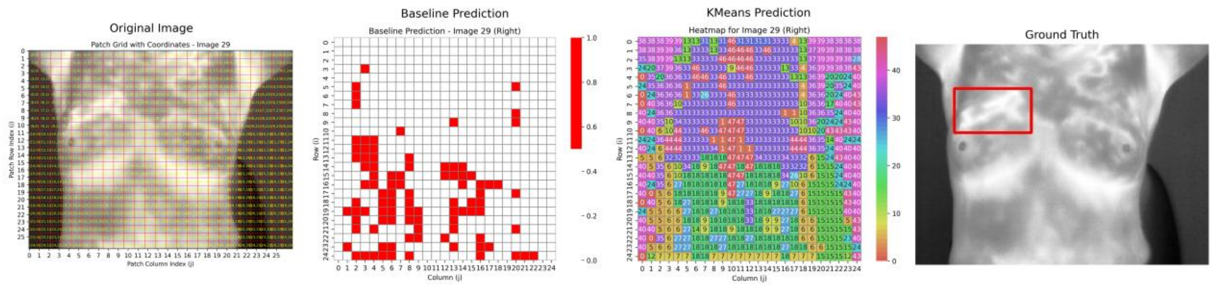
Original Image
Patch Grid with Coordinates - Image 24
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 24 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 24 (Left)
Row (i)
Column (j)
Ground Truth

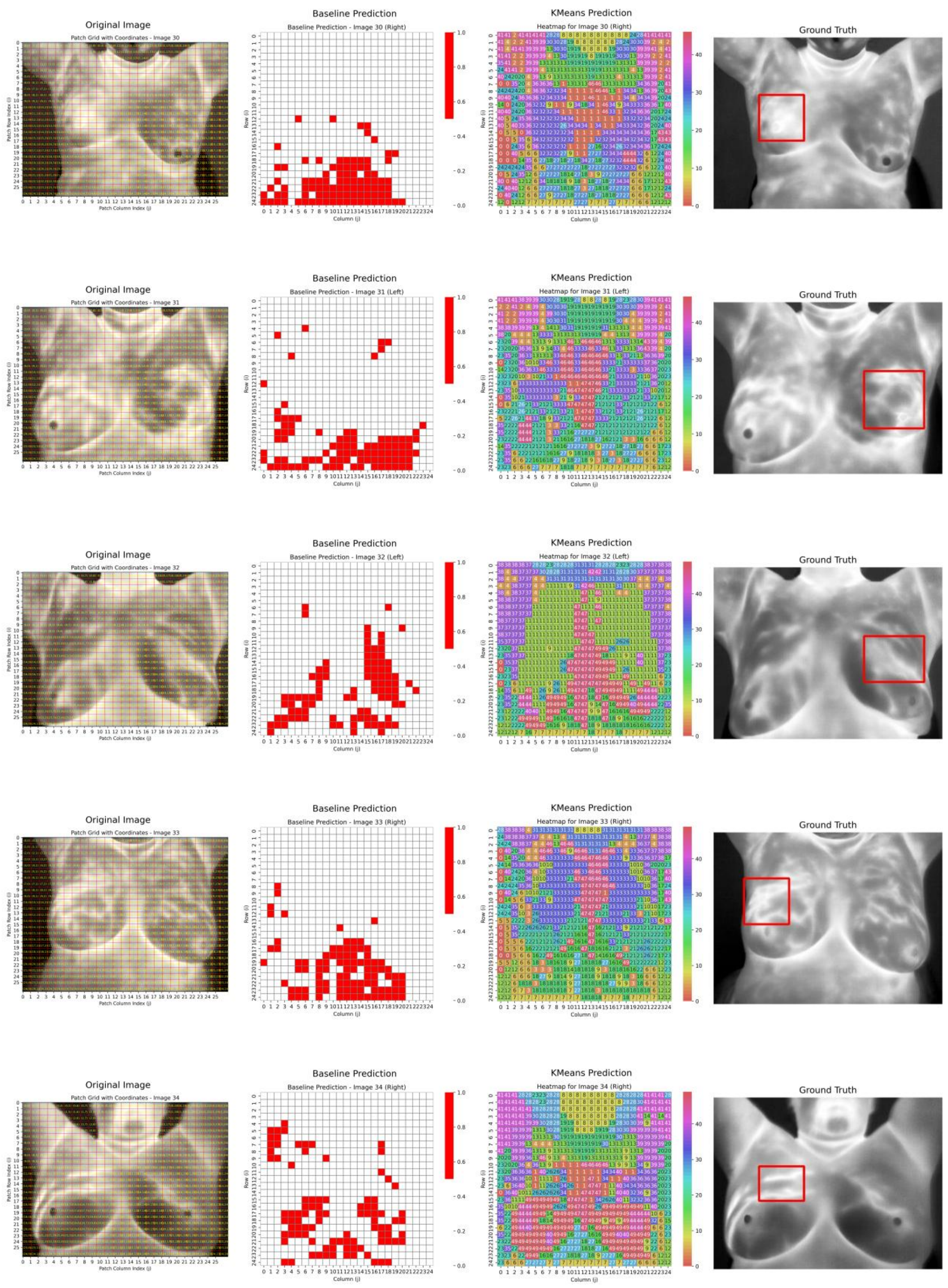

Original Image
Patch Grid with Coordinates - Image 25
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 25 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 25 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 26
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 26 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 26 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 27
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 27 (Left)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 27 (Left)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 28
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 28 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 28 (Right)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 29
Patch Row Index (i)
Patch Column Index (j)
Baseline Prediction
Baseline Prediction - Image 29 (Right)
Row (i)
Column (j)
KMeans Prediction
Heatmap for Image 29 (Right)
Row (i)
Column (j)
Ground Truth


Original Image
Patch Grid with Coordinates - Image 30
Baseline Prediction
Baseline Prediction - Image 30 (Right)
KMeans Prediction
Heatmap for Image 30 (Right)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 31
Baseline Prediction
Baseline Prediction - Image 31 (Left)
KMeans Prediction
Heatmap for Image 31 (Left)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 32
Baseline Prediction
Baseline Prediction - Image 32 (Left)
KMeans Prediction
Heatmap for Image 32 (Left)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 33
Baseline Prediction
Baseline Prediction - Image 33 (Right)
KMeans Prediction
Heatmap for Image 33 (Right)
Ground Truth
Original Image
Patch Grid with Coordinates - Image 34
Baseline Prediction
Baseline Prediction - Image 34 (Right)
KMeans Prediction
Heatmap for Image 34 (Right)
Ground Truth
Patch Row Index (i)
Patch Column Index (j)
Row (i)
Column (j)

Plot 3:Comparison with 25X25 original image, baseline, k-means prediction using the experiment 5.1.2.3 with raw features and the GT

5.2.3 Testing trained model on healthy women

For the raw features I was looking for the cluster 3 and unfortunately the 7 from 10 images have the cluster 3 and its mostly in the breast area so this example did not work as it is supposed to [Figure 18]. For the reduced features I am looking for the cluster 18 and again 6 from 10 images have the tumour cluster but this time the patches on the cluster are mostly above the breast area and they have big quantity, so the result in this experiment is better but need a human interaction to understand that the tumour clusters is not detecting a tumour [Figure 19].

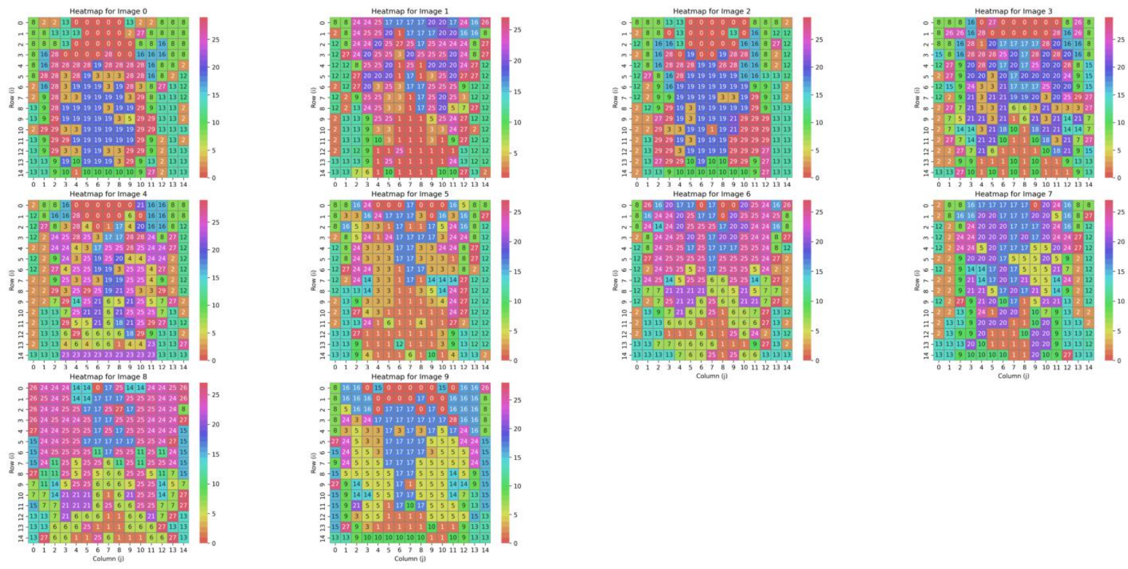


Figure 18: An example of 10 heatmaps from thermographs with health women with raw features and 30 clusters

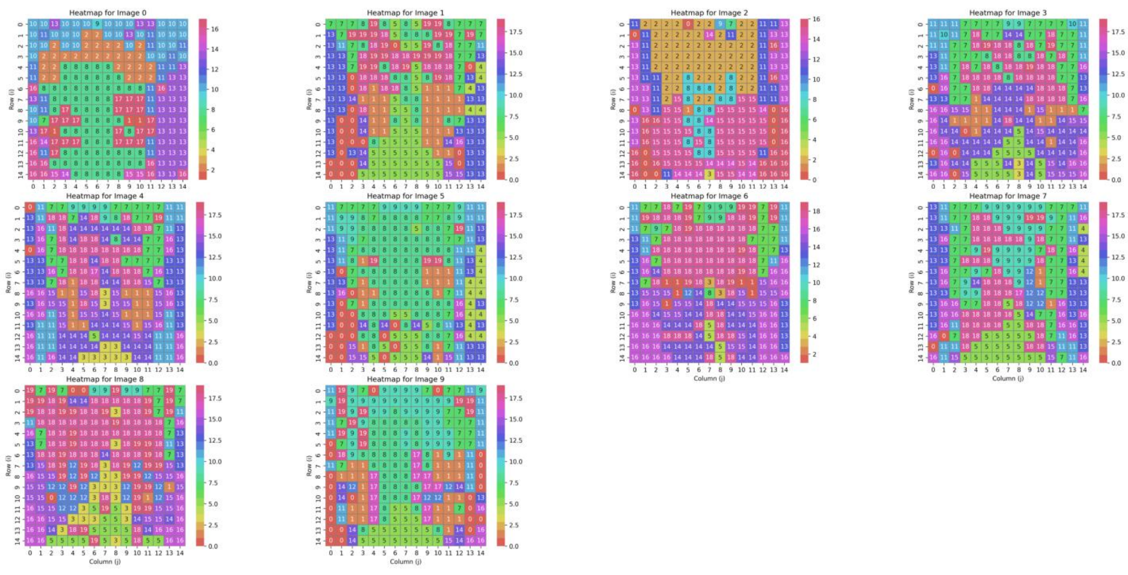


Figure 19: An example of 10 heatmaps from thermographs with health women with raw features and 30 clusters

5.2.4 Supervised on unsupervised training

First, I tried this model with 0.6 [Figure 21] and 0.7 [Figure 22] probability threshold but in both trials the results are wrong because of the poor labeled data, the imbalances of the labels (the most tumor areas were in the same breast area) and the unprofessional detection of the cancer area. Based on the example of the 10 images [Figure 20] we can see that with 0.6 probability threshold [Figure 21] there are some images that did not match with any label and some matched with two labels. In the other hand with 0.7 probability threshold [Figure 22] only one of the 10 image matched with 2 labels but most of the images matched with non.

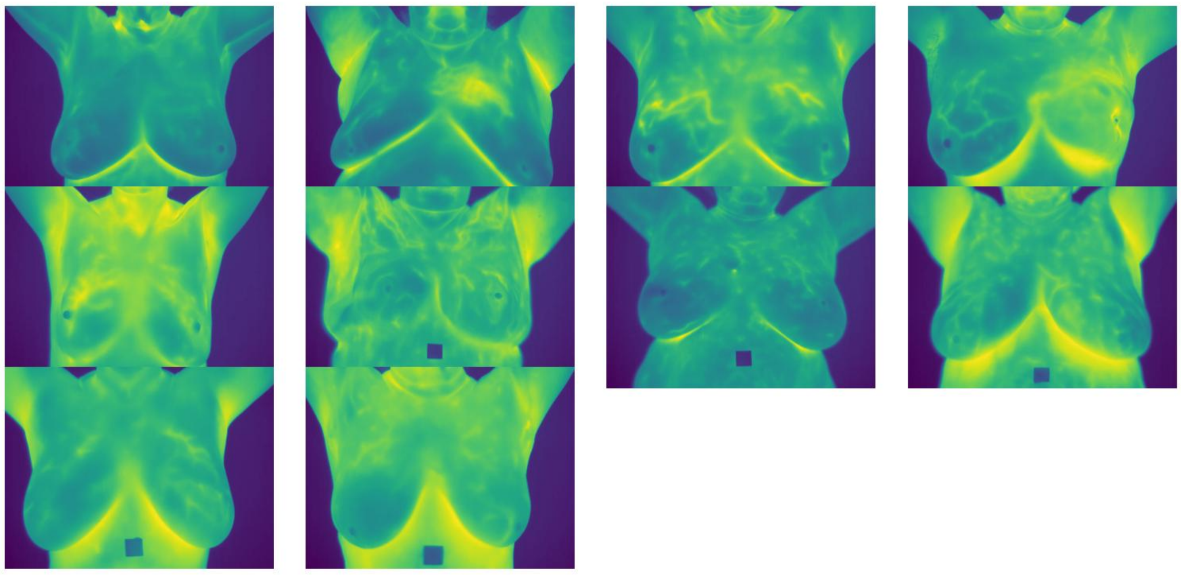


Figure 20: An example of 10 thermographs with sick women that I used for Supervised on unsupervised training

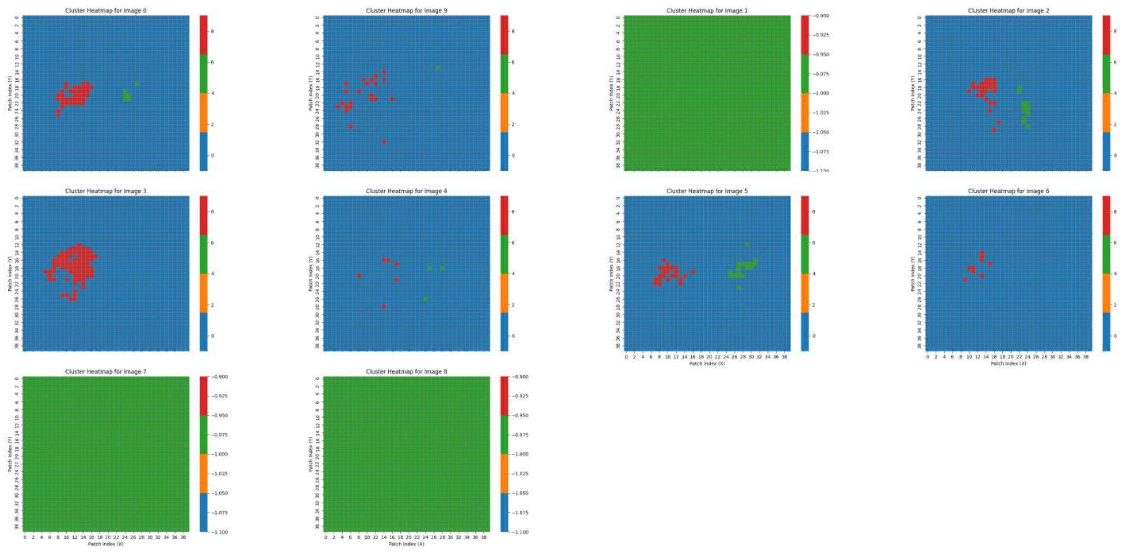


Figure 21: The 10 heatmaps from the thermographs with sick women that I used for Supervised on unsupervised training with 0.6 probability threshold

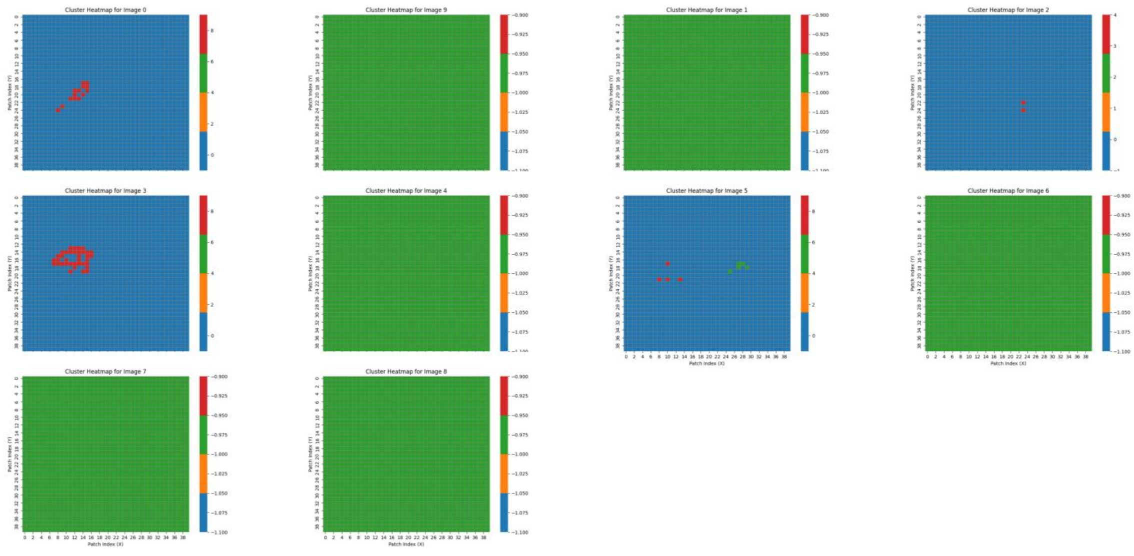


Figure 22: The 10 heatmaps from the thermographs with sick women that I used for Supervised on unsupervised training with 0.6 probability threshold

Chapter 6

Conclusion & Future Work

6.1 Conclusion

The purpose of my thesis was to develop un unsupervised model that detect the location of the breast cancer on thermographs of ill women to increase the number of early detections of breast cancer.

More specific I mostly pre-processed the images with dinov2 and try to apply different ideas on the k-means unsupervised classifier. In my thesis I did not get very high result, but I tried many ideas [Table 4] with a high potential to work match better with more data , better image pre-processing and using qualified labels for training and evaluation.

Experiment	Cluster	Precision	Recall	F1 Score	Mean AP
15X15 Raw Features	20	12%	40%	19%	24%
	25	14%	36%	20%	25%
	30	15%	31%	20%	29%
15X15 PCA Feature reduction to 128	20	14%	37%	20%	24%
	25	13%	35%	19%	24%
	30	13%	46%	20%	38%
15X15 Raw Features with tumour cluster reassign using K-means centres	20	13%	29%	18%	36%
	25	16%	25%	20%	37%

	30	18%	24%	20%	41%
15X15 PCA Feature reduction to 128 with cluster reassign using K-means centres	20	17%	26%	21%	37%
	25	16%	25%	20%	36%
	30	13%	40%	19%	46%
15X15 Raw Features with tumour cluster reassign using Euclidean distance	20	15%	32%	20%	37%
	25	20%	30%	24%	38%
	30	21%	27%	24%	42%
15X15 PCA Feature reduction to 128 with tumour cluster reassign using Euclidean	20	21%	32%	25%	38%
	25	18%	28%	22%	37%
	30	17%	22%	19%	41%
15X15 Using more data with raw features	20	13%	27%	17%	29%
	25	15%	20%	17%	39%
	30	16%	32%	21%	30%
15X15 Using more data with PCA Feature reduction to 128	20	14%	37%	20%	29%
	25	13%	27%	18%	29%
	30	14%	26%	18%	33%
15X15 Raw features with patch selection	20	15%	35%	21%	26%
	25	16%	20%	18%	43%
	30	22%	25%	23%	45%
15X15 PCA Feature reduction to 128 with patch selection	20	14%	33%	20%	27%
	25	15%	28%	20%	32%
	30	17%	22%	19%	41%
15X15 Patch selection and tumour cluster reassign using Euclidean	30 raw	28%	22%	25%	56%
	20 reduced	19%	28%	23%	40%
25X25 Raw features	50	20%	21%	21%	47%
25X25 PCA Feature reduction to 128	30	17%	34%	23%	31%
	50	19%	22%	21%	44%
25X25 Tumour cluster reassign using Euclidean distance	50 raw	26%	18%	21%	59%
	50 reduced	24%	18%	20%	56%
25X25 Patch selection	50 raw	18%	13%	15%	57%
	50 reduced	21%	19%	20%	52%

Table 4: Experiments and evaluation

In my experiments the models were mostly successfully detecting the tumour area in images where the area was obviously brighter than the other areas. But over all with the labels I created the best results were with 15X15 patches, raw features , patch reassigning with the Euclidean distance, patch selection and 30 clusters where I manage to get 28% precision 22% recall and 25% f1 score. Also, the best results with reduced features to 128 were with 15X15 patches, patch reassigning with Euclidean distance and 20 cluster and I was able to get 21% precision 32% recall and 25% f1 score. Lastly I managed to get the best mean AP with 59% using 25X25 patches, cluster reassign using Euclidean distance and 50 clusters on raw features.

6.2 Future Work

The ideas I tried in my model have the potential to produce better results than the ones I got. I believe that I created the bone for a model that can work better with some changes. So, the next phase will be to try the models with more data and more qualified labels for the data or a suggestion was given to me to proses the images with sodu colouring for better visibility of the cancer area. Also is possible to make changes in the images pre-processing for better results and try different unsupervised model that may work better with this data and even different number of pca feature reduction may give better results.

References

[1] "Key Statistics for Breast Cancer," American Cancer Society, Jan. 22, 2025. [Online]. Available: https://www.cancer.org/cancer/types/breast-cancer/about/how-common-is-breast-cancer.html

[2] European Commission Initiative on Breast Cancer (ECIBC), *European Guidelines on Breast Cancer Screening and Diagnosis: Recommendations List*, European Union, Apr. 2025. [Online]. Available: https://healthcare-quality.jrc.ec.europa.eu/ecibc/european-breast-cancer-guidelines

[3] M. Pafitis, *MammoCheck: An affordable and safe method for breast cancer examination at home*, M.S. thesis, Frederick Univ., 2022.

[4] U.S. Food and Drug Administration, "FDA issues warning letter to clinic illegally marketing unapproved thermography device, warns consumers to avoid using thermography devices to detect breast cancer," Feb. 25, 2019. [Online]. Available: https://www.fda.gov/news-events/press-announcements/fda-issues-warning-letter-clinic-illegally-marketing-unapproved-thermography-device-warns-consumers

[5] A. D. Kulkarni and B. Lowe, "Random Forest Algorithm for Land Cover Classification," *Comput. Sci. Fac. Publ. Present.*, Paper 1, 2016. [Online]. Available: http://hdl.handle.net/10950/341

[6] A. Chaudhary, S. Kolhe, and R. Kamal, "An improved random forest classifier for multi-class classification," *School of Computer Science and IT*, Devi Ahilya University, Indore, India. Volume 3, Issue 4, December 2016, Pages 215-222

[7] T. M. Kodinariya and P. R. Makwana, "Review on determining number of clusters in K-means clustering," *Int. J. Adv. Res. Comput. Sci. Manag. Stud.*, vol. 1, no. 6, pp. 90–95, June 2013. [Online]. Available: www.ijarcsms.com

[8] G. Ayana *et al.*, "Vision-transformer-based transfer learning for mammogram classification," *Diagnostics*, vol. 13, no. 2, p. 178, 2023, doi: 10.3390/diagnostics13020178.

[9] M. Oquab *et al.*, "DINOv2: Learning Robust Visual Features without Supervision," *Trans. Mach. Learn. Res.*, 2024. [Online]. Available: https://arxiv.org/abs/2304.07193.

[10] A. Gunawardana and G. Shani, "A survey of accuracy evaluation metrics of recommendation tasks," *J. Mach. Learn. Res.*, vol. 10, pp. 2935–2962, Dec. 2009.

[11] M. Z. Naser and A. H. Alavi, "Insights into performance fitness and error metrics for machine learning," unpublished manuscript, Glenn Dept. of Civil Eng., Clemson Univ., Clemson, SC, and Dept. of Civil and Environ. Eng., Univ. of Pittsburgh, Pittsburgh, PA, 2023.

[12] S. Sathyanarayanan and B. R. Tantri, "Confusion Matrix-Based Performance Evaluation Metrics," *Afr. J. Biomed. Res.*, vol. 27, no. 4s, pp. 4023–4031, Nov. 2024, doi: 10.53555/AJBR.v27i4S.4345.

[13] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A survey of modern deep learning based object detection models," *Digital Signal Processing*, vol. 126, p. 103514, 2022, doi: 10.1016/j.dsp.2022.103514.

[14] R. Lawson, "Implications of surface temperatures in the diagnosis of breast cancer," *Can. Med. Assoc. J.*, vol. 75, no. 4, pp. 309–310, Aug. 1956.

[15] J.-L. Gonzalez-Hernandez *et al.*, "Technology, application and potential of dynamic breast thermography for the detection of breast cancer," *Int. J. Heat Mass Transf.*, vol. 131, pp. 558–573, 2019, doi: 10.1016/j.ijheatmasstransfer.2018.11.089.

[16] S. Rodriguez-Guerrero *et al.*, “Breast Thermography,” *Mendeley Data*, V3, 2024. [Online]. Available: https://doi.org/10.17632/mhrt4svjxc.3

[17] L. F. Silva *et al.*, "A new database for breast research with infrared image," *J. Med. Imaging Health Inform.*, vol. 4, no. 1, pp. 92–100, 2014.

Appendix A

Codes

Dinov2 pre-processing:

```
 # Imports for data manipulation and processing
import torch
import torchvision.transforms as T
import numpy as np
from sklearn.decomposition import PCA
import sklearn
# Imports for data loading and visualization
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
# Other Utility Imports
import os

#Mount the google driver
from google.colab import drive
drive.mount('/content/drive')

# Path to your images folder in Google Drive
image_folder = "/content/drive/My Drive/{image folder}"

# Generate image paths using actual filenames
img_paths = [os.path.join(image_folder, filename) for filename in
sorted(os.listdir(image_folder)) if filename.endswith(".jpg")]
print(img_paths)

##### EXPERIMENT PARAMETERS #####
N_IMGS =  35 # Number of images of the specific class to work with
# Plotting Parameters
N_ROW_IMGS = 6 # How many rows of images to show in the plotting
```

```
N_COL_IMGS = 6 # How many of those images will appear in each row

# Define the number of patches along the two dimensions to split each image into
patch_h = 15
patch_w = 15
# Define the size of the model to use for extracting the features of each patch
model_size = 's' # options: s, b, l, g
use_registers = False
use_extended_dinov2 = True

# Instructions:
# From https://github.com/vpariza/NeCo?tab=readme-ov-file download the student Dinov2
ViT-S/14 from  https://1drv.ms/u/c/67fac29a77adbae6/EWvXdau9r6NIr-
vIc_xDlxAB1sDrljoaPR_A3JhIEeE8dw?e=pOXEXG
# upload to google drive
# and find the path to the file on google drive
path = "/content/drive/MyDrive/images/neco_on_dinov2r_vit14_model.ckpt"
!ls /content/drive/MyDrive/images/neco_on_dinov2r_vit14_model.ckpt



model = torch.hub.load('facebookresearch/dinov2', 'dinov2_vits14')
model.load_state_dict(torch.load(path), strict=False)

tensor = torch.randn(N_IMGS, 3, 224, 224) # 16*14 =224
display(tensor.shape)
if use_registers:
  model = torch.hub.load('facebookresearch/dinov2',
'dinov2_vit{}14_reg'.format(model_size))
else:
  model = torch.hub.load('facebookresearch/dinov2', 'dinov2_vit{}14'.format(model_size))
features_dict = model.forward_features(tensor)
display(features_dict.keys())
display(features_dict['x_norm_clstoken'].shape)
display(features_dict['x_norm_regtokens'].shape)
display(features_dict['x_norm_patchtokens'].shape)
display(features_dict['x_prenorm'].shape)
```

```
feat_dims = {
    's':384, # vits14
    'b':768, # vitb14
    'l':1024, # vitl14
    'g':1536, # vitg14
}
feat_dim = feat_dims[model_size]

transform = T.Compose([
    T.GaussianBlur(9, sigma=(0.1, 2.0)),
    T.Resize((patch_h * 14,patch_w * 14), interpolation=T.InterpolationMode.LANCZOS),
    T.CenterCrop((patch_h * 14, patch_w * 14)),
    T.ToTensor(),
    T.Normalize(mean=(0.485, 0.456, 0.406), std=(0.229, 0.224, 0.225)),
    # T.Normalize(mean=0.5, std=0.2),
])

if use_registers:
  model = torch.hub.load('facebookresearch/dinov2',
'dinov2_vit{}14_reg'.format(model_size))
else:
  model = torch.hub.load('facebookresearch/dinov2', 'dinov2_vit{}14'.format(model_size))

if use_extended_dinov2 == False:
  model.load_state_dict(torch.load(path), strict=False)

# extract DINOv2 features
features = torch.zeros(N_IMGS, patch_h * patch_w, feat_dim)
imgs_tensor = torch.zeros(N_IMGS, 3, patch_h * 14, patch_w * 14)
for i in range(N_IMGS):
    img_path = img_paths[i]
    img = Image.open(img_path).convert('RGB')
    imgs_tensor[i] = transform(img)[:3]
with torch.no_grad():
  if torch.cuda.is_available():
    features_dict = model.cuda().forward_features(imgs_tensor.cuda())
  else:
    features_dict = model.forward_features(imgs_tensor)
  print(features_dict.keys())
  features = features_dict['x_norm_patchtokens'].cpu()


save_path = "/content/drive/MyDrive/{feature file}"  # Change this
import torch

torch.save(features, save_path)  # Save as a PyTorch tensor
print("Features saved successfully!")
```

K-means:

```
import torch
from sklearn.cluster import KMeans
import numpy as np

#Check the shape
# print("Loaded features shape:", features.shape)
feature_files = [
    {dinov2 feature files}
]

# Load and concatenate features
features_list = [torch.load(f) for f in feature_files]

for i, f in enumerate(features_list):
    print(f"File {feature_files[i]} -> Shape: {f.shape}")

#Feature load with 15X15 pacthes
(
NUM_PATCHES=int(features_list[0].shape[0]/35)
FEATURE_DIM = features_list[0].shape[1]
N_IMGS=int(features_list[0].shape[0]/225)
features_list[0]= features_list[0].reshape(N_IMGS, NUM_PATCHES,FEATURE_DIM)
features=features_list[0]
)

# Concatenate along the batch dimension (axis=0)
features = torch.cat(features_list, dim=0)

# Check the new shape
print("Loaded combined features shape:", features.shape)



# Assuming features shape: (N_IMGS, NUM_PATCHES, FEATURE_DIM)
N_IMGS,NUM_PATCHES, FEATURE_DIM = features.shape
```

```
# Reshape to cluster patches instead of entire images
patch_features_flattened = features.reshape(-1, FEATURE_DIM)  # Shape: (N_IMGS * NUM_PATCHES, FEATURE_DIM)

from sklearn.decomposition import PCA

pca = PCA(n_components=128) # Reduce feature dimension
features_reduced = pca.fit_transform(patch_features_flattened)



# Apply K-Means to patches
K = 30 # Number of clusters

# Run KMeans on data
kmeans = KMeans(n_clusters=K, random_state=42, n_init=10)
patch_clusters = kmeans.fit_predict({features_reduced/ patch_features_flattened) #reduced or raw



# Reshape back to (N_IMGS, NUM_PATCHES) to know which patch belongs to which cluster
patch_clusters = patch_clusters.reshape(N_IMGS, NUM_PATCHES)

print("Patch Clusters Shape:", patch_clusters.shape)  # Should be (N_IMGS, NUM_PATCHES)
print(patch_clusters)

patch_positions = np.zeros((N_IMGS, NUM_PATCHES, 2), dtype=int)  # Store (i, j) positions



for img_idx in range(N_IMGS):
    for patch_idx in range(NUM_PATCHES):
        i = patch_idx // 40  # Row index
        j = patch_idx % 40  # Column index
        patch_positions[img_idx, patch_idx] = (i, j)  # Store position



import os
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Define a color palette for more than 20 clusters
cmap = sns.color_palette("hls", K)



# # Define labels for each heatmap
labels = [
    "Left", "Left", "Right", "Right", "Left", "Left", "Left", "Right", "Right", "Right",
    "Left", "Right", "Right", "Right", "Left", "Right", "Right", "Left", "Left", "Left",
    "Right", "Right", "Left", "Right", "Left", "Left", "Left", "Left", "Right", "Right",
    "Right", "Left", "Left", "Right", "Right"
]  # Ensure this matches N_IMGS

# Define the output folder on the desktop
desktop_path = os.path.join(os.path.expanduser("~"), "Desktop")
output_folder = os.path.join(desktop_path, "Heatmap_Images_40_K20")

# Create folder if it doesn't exist
os.makedirs(output_folder, exist_ok=True)

# Loop through images and generate heatmaps
for img_idx in range(N_IMGS):
    heatmap = np.full((40, 40), -1)  # Initialize with -1 (for missing values)


    for patch_idx in range(NUM_PATCHES):
        i, j = patch_positions[img_idx, patch_idx]
        cluster_id = patch_clusters[img_idx, patch_idx]
        heatmap[i, j] = cluster_id  # Assign cluster ID to heatmap position
    # Create the heatmap figure
    plt.figure(figsize=(8, 6))
    # Create annotation labels, blank for -1
    annot_labels = np.where(heatmap == -1, "", heatmap.astype(int).astype(str))

    sns.heatmap(
     heatmap,
     cmap=cmap,
     annot=annot_labels,
     fmt="",
     linewidths=0.5,
     linecolor='gray',
     cbar=True,
     square=True
    )

    # Define the filename using the corresponding label
    label = labels[img_idx] if img_idx < len(labels) else "Unknown"
    filename = f"heatmap_{img_idx}_{label}.png"
    save_path = os.path.join(output_folder, filename)

    # Save the heatmap
    plt.title(f"Heatmap for Image {img_idx} ({label})")
    plt.xlabel("Column (j)")
    plt.ylabel("Row (i)")
    plt.savefig(save_path, dpi=300, bbox_inches="tight")
    plt.close()  # Close the figure to free memory

print(f"Heatmaps saved in: {output_folder}")

# evaluation
import csv
import numpy as np

# Load your CSV into a numpy array (excluding the first index column)
csv_file = "output_40.csv"  # ← change this to your actual CSV file name
```

```
with open(csv_file, "r") as file:
    reader = csv.reader(file)
    csv_data = np.array([list(map(int, row[1:])) for row in reader])  # Skip first column

print("CSV shape:", csv_data.shape)  # Should be (N_IMGS, 225)

TP = 0  # CSV == 1 and Cluster == {tumour cluster}
TN = 0  # CSV == 0 and Cluster != {tumour cluster}
FP = 0  # CSV == 0 and Cluster == {tumour cluster}
FN = 0  # CSV == 1 and Cluster != {tumour cluster}

for img_idx in range(35):
    for patch_idx in range(NUM_PATCHES):
        cluster = patch_clusters[img_idx, patch_idx]
        label = csv_data[img_idx, patch_idx]

        if label == 1 and (cluster == {tumour cluster} or  cluster == {tumour cluster}):
            TP += 1
        elif label == 0 and (cluster != {tumour cluster} and cluster != {tumour cluster}):
            TN += 1
        elif label == 0 and (cluster == {tumour cluster} or cluster == {tumour cluster}):
            FP += 1
        elif label == 1 and (cluster != {tumour cluster} and cluster != {tumour cluster}):
            FN += 1

total = TP + TN + FP + FN
accuracy = (TP + TN) / total
precision = TP / (TP + FP) if (TP + FP) > 0 else 0
recall = TP / (TP + FN) if (TP + FN) > 0 else 0
f1 = 2 * (precision * recall) / (precision + recall) if (precision + recall) > 0 else 0

print(f"✅ Accuracy:  {accuracy:.4f}")
print(f"🎯 Precision: {precision:.4f}")
print(f"📈 Recall:    {recall:.4f}")
print(f"💡 F1 Score:  {f1:.4f}")
```

Labelling for supervised:

```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as patches
import matplotlib.image as mpimg
import torch
import pandas as pd
import os

# --- Load Features ---
feature_files = [{feature files}]
features_list = [torch.load(f) for f in feature_files]
features = torch.cat(features_list, dim=0)

N_IMGS, NUM_PATCHES, FEATURE_DIM = features.shape
#For 15X15 patches
(
# NUM_PATCHES=int(features.shape[0]/35)
# FEATURE_DIM = features.shape[1]
# N_IMGS=int(features.shape[0]/225)
)

# --- Load Image Paths ---
img_folder = "C:/Users/damia/Desktop/sick3" #add you images' folder
img_paths = sorted([os.path.join(img_folder, f) for f in os.listdir(img_folder) if
f.endswith(('.png', '.jpg', '.jpeg'))])

# --- Dynamically Determine Grid Size ---
PATCH_H = int(np.sqrt(NUM_PATCHES))
PATCH_W = int(np.sqrt(NUM_PATCHES))



# --- CSV File ---
save_path = "labeled_patch_features.csv"
```

```
if not os.path.exists(save_path):
    df = pd.DataFrame(columns=["Image_Index", "Patch_Index", "Label"] + [f"Feature_{i}"
for i in range(FEATURE_DIM)])
    df.to_csv(save_path, index=False)


# --- Function to Select Region Manually ---
def on_click(event):
    global large_patch_x, large_patch_y, large_patch_w, large_patch_h
    if event.xdata is None or event.ydata is None:
        return
    if 'start' not in on_click.__dict__:
        on_click.start = (event.xdata, event.ydata)
    else:
        large_patch_x, large_patch_y = int(on_click.start[0]), int(on_click.start[1])
            large_patch_w, large_patch_h = int(event.xdata - large_patch_x), int(event.ydata -
large_patch_y)
        plt.close()


# --- Manual Image Selection Loop ---
while True:
    print("\nAvailable images:")
    for i, path in enumerate(img_paths):
        print(f"{i}: {os.path.basename(path)}")

    try:
        img_idx = int(input("\nEnter image number to process (or -1 to exit): "))
        if img_idx == -1:
            print("Exiting program. ✅ ")
            break
        if img_idx < 0 or img_idx >= len(img_paths):
            print("Invalid number. Please try again.")
            continue
    except ValueError:
        print("Invalid input. Please enter a number.")
        continue

    img_path = img_paths[img_idx]
    img = mpimg.imread(img_path)
    img_h, img_w = img.shape[:2]  # Get actual image size


    # Dynamically determine patch size based on image size
    patch_w = img_w // PATCH_W
    patch_h = img_h // PATCH_H

    print(f"\nProcessing Image {img_idx}: {img_path}")
    print(f"Image size: {img_w}x{img_h}, Patch size: {patch_w}x{patch_h}, Grid: {PATCH_W}x{PATCH_H}")


    # --- Show Image for Manual Selection ---
    fig, ax = plt.subplots()
    ax.imshow(img)
    ax.set_title("Click to select a region (drag from start to end)")
    fig.canvas.mpl_connect("button_press_event", on_click)
    plt.show()


    # Ensure width and height are positive
    large_patch_w = abs(large_patch_w)
    large_patch_h = abs(large_patch_h)


    # --- Find Small Patches Inside the Selected Region ---
    selected_patches = []
    for patch_idx in range(NUM_PATCHES):
        patch_row = patch_idx // PATCH_W
        patch_col = patch_idx % PATCH_W

        patch_x = patch_col * patch_w
        patch_y = patch_row * patch_h

        if (large_patch_x <= patch_x < large_patch_x + large_patch_w) and (
            large_patch_y <= patch_y < large_patch_y + large_patch_h
        ):
            selected_patches.append(patch_idx)

    print(f"Selected patches: {selected_patches}")



    # --- Show Image with Selected Area ---
    fig, ax = plt.subplots()
    ax.imshow(img)
    rect = patches.Rectangle((large_patch_x, large_patch_y), large_patch_w, large_patch_h,
                             linewidth=2, edgecolor='r', facecolor='none')
    ax.add_patch(rect)

    for patch_idx in selected_patches:
        patch_row = patch_idx // PATCH_W
        patch_col = patch_idx % PATCH_W
        patch_x = patch_col * patch_w
        patch_y = patch_row * patch_h
        patch_rect = patches.Rectangle((patch_x, patch_y), patch_w, patch_h,
                                       linewidth=1, edgecolor='b', facecolor='none')
        ax.add_patch(patch_rect)

    ax.set_title(f"Selected Region - Image {img_idx}")
    plt.show()



    # --- Assign a Label ---
     label = input(f"Enter label for selected region in Image {img_idx} (e.g., 'arm', 'chest',
'background'): ")



    # --- Store Labeled Features ---
    labeled_patches = []
    for patch_idx in selected_patches:
        feature_vector = features[img_idx, patch_idx, :].numpy().tolist()
        labeled_patches.append([img_idx, patch_idx, label] + feature_vector)



    # --- Append to CSV ---
    df_new = pd.DataFrame(labeled_patches, columns=["Image_Index", "Patch_Index",
"Label"] + [f"Feature_{i}" for i in range(FEATURE_DIM)])
    df_new.to_csv(save_path, mode="a", header=False, index=False)

    print(f"✅ {len(selected_patches)} labeled patches from Image {img_idx} added to
{save_path}.")
    print("\nYou can select another image or exit.")
```

Random Forest

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder

# Load the labeled CSV
df = pd.read_csv("labeled_patch_features3.csv")

# Extract features and labels from the CSV
X_csv = df.iloc[:, 3:].values  # Features (ignore Image_Index, Patch_Index, Label)
y_csv = df["Label"].values      # Labels

# Encode labels into numerical values
label_encoder = LabelEncoder()
y_csv_encoded = label_encoder.fit_transform(y_csv)

# Get indices of patches that belong to breast Cluster (cluster 0 in this example)
breast_cluster_indice= np.where(np.isin(patch_clusters, [0]))
num_patches_in_breast_cluster= breast_cluster_indices [0].shape[0]  # Count of selected patches
print("Number of patches in clusters :", num_patches_in_breast_cluster)

# Extract features of these patches
breast_cluster_indices_flat = breast_cluster_indices [0] * NUM_PATCHES +
breast_cluster_indices [1]
breast_cluster_features = patch_features_flattened[breast_cluster_indices _flat]

# Split CSV data for training
X_train, X_test, y_train, y_test = train_test_split(X_csv, y_csv_encoded, test_size=0.2, random_state=42)
```

```
# Train a classifier on the labeled CSV data
clf = RandomForestClassifier(n_estimators=100,random_state=42)
clf.fit(X_train, y_train)

print("Shape of breast_cluster_features:", breast_cluster_features.shape)



from collections import Counter
print("Training Label Distribution:", Counter(y_train))

#Set the threshold
pobability_threshold=0.7

probs = clf.predict_proba(breast_cluster_features)  # shape: (n_patches, n_classes)
max_probs = np.max(probs, axis=1)
best_class_ids = np.argmax(probs, axis=1)



# Filter patches with confident prediction
confident = max_probs >= pobability_threshold
confident_patch_features = breast_cluster_features[confident]
confident_indices = breast_cluster_indices _flat[confident]
confident_labels = best_class_ids[confident]
confident_labels_decoded = label_encoder.inverse_transform(confident_labels)
# Initialize the matrix with -1s (unlabeled)
new_patch_clusters = np.full_like(patch_clusters, fill_value=-1)



# Assign predicted labels to their corresponding [img_idx, patch_idx] positions
for idx, flat_idx in enumerate(confident_indices):
    img_idx, patch_idx = divmod(flat_idx, NUM_PATCHES)  # Convert flat index back to
2D index
   new_patch_clusters[img_idx, patch_idx] = confident_labels[idx]  # Store the class label
```

```
#Display of heatmap

import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np


# Define a colormap for different clusters
cmap = sns.color_palette("tab10", np.unique(new_patch_clusters).size)  # Choose 10 distinct
colors


# Function to plot heatmap for a given image index
def plot_cluster_heatmap(image_idx):
    plt.figure(figsize=(8, 6))

     print("Shape of patch_clusters:", new_patch_clusters.shape)  # Should be (num_images,
height, width)
    # Ensure it's a 2D array
    # Reshape cluster_map to 2D (assuming square patches)
    GRID_HEIGHT = 40  # Adjust based on actual number of patches per row
    GRID_WIDTH = 40   # Adjust based on actual number of patches per column

    cluster_map = new_patch_clusters[image_idx].reshape(GRID_HEIGHT, GRID_WIDTH)


   # Check shape before plotting
    print(f"Reshaped cluster map shape for image {image_idx}: {cluster_map.shape}")


    # Check shape before plotting
    print(f"Cluster map shape for image {image_idx}: {cluster_map.shape}")
    # Plot heatmap
      sns.heatmap(cluster_map,  cmap=cmap,  linewidths=0.5,  linecolor="gray",  annot=False,
cbar=True)

    plt.title(f"Cluster Heatmap for Image {image_idx}")
    plt.xlabel("Patch Index (X)")
    plt.ylabel("Patch Index (Y)")
    plt.show()


# Example: Plot heatmap for a specific image (change index as needed)
plot_cluster_heatmap(image_idx=0)  # Change 0 to any image index


for i in range(N_IMGS):  # Replace with actual number of images
    plot_cluster_heatmap(i)
```

Labelling for evaluation

```
import csv

def generate_csv_from_input(filename, rows=35, cols=1600):
    with open(filename, mode='w', newline='') as file:
        writer = csv.writer(file)
        for i in range(1, rows + 1):
            print(f"\nRow {i}:")
            count = int(input(f"  How many 1's do you want in this row? "))

            # Loop until valid input
            while True:
                indices_input = input(f"  Enter {count} column numbers (1–{cols}) separated by spaces: ")
                try:
                    indices = list(map(int, indices_input.strip().split()))
                    if len(indices) != count or not all(1 <= idx <= cols for idx in indices):
                        raise ValueError
                    break
                except ValueError:
                    print("  ❌ Invalid input. Please enter exactly the correct number of integers between 1 and 225.")

            # Initialize row with index and 0s
            row = [i] + [0] * cols
            # Set selected indices to 1
            for idx in indices:
                row[idx] = 1  # Index 1 refers to the first data column (after the index)

            # Write row to CSV
            writer.writerow(row)
    print("\n✅ CSV file has been generated.")
# Usage
generate_csv_from_input("output.csv")
```

Addition code for K-means to pass additional image through the model:

```
# Setup heatmap output
desktop_path = os.path.join(os.path.expanduser("~"), "Desktop")
output_folder = os.path.join(desktop_path, "{Heatmap_Images}") # Replace with your file
os.makedirs(output_folder, exist_ok=True)

# Load multiple new images from a .pt file
new_image_file = "{feature file}"  # Replace with your file
new_images_features = torch.load(new_image_file) # Shape: (N_NEW_IMGS, NUM_PATCHES, FEATURE_DIM)
print(new_images_features.shape)
# Get the number of new images
N_NEW_IMGS =int(new_images_features.shape[0])

# Color map
cmap = sns.color_palette("hls", K)

# Loop through new images
for img_idx in range(N_NEW_IMGS):
    # Extract patch features for this image
    new_image_features_flat = new_images_features.reshape(-1, FEATURE_DIM)
    #PCA reduction if wanted
    new_features_reduced = pca.fit_transform(new_image_features_flat)

    # Predict cluster IDs
    new_image_clusters = kmeans.predict(new_features_reduced)

   #  Reshape back to (N_IMGS, NUM_PATCHES) to know which patch belongs to which cluster
    new_image_clusters = new_image_clusters.reshape(N_NEW_IMGS, NUM_PATCHES)

    # Build heatmap matrix
    heatmap = np.full((15, 15), -1)  # Adjust size as needed
```

```
        for patch_idx in range(NUM_PATCHES):
            i, j = patch_positions[img_idx, patch_idx]
            cluster_id = new_image_clusters[img_idx,patch_idx]
            heatmap[i, j] = cluster_id

        annot_labels = np.where(heatmap == -1, "", heatmap.astype(int).astype(str))

        sns.heatmap(
         heatmap,
         cmap=cmap,
         annot=annot_labels,
         fmt="",
         linewidths=0.5,
         linecolor='gray',
         cbar=True,
         square=True
        )



        plt.title(f"Heatmap for Image {img_idx}")
        plt.xlabel("Column (j)")
        plt.ylabel("Row (i)")

        filename = f"heatmap_{img_idx}.png"
        save_path = os.path.join(output_folder, filename)
        plt.savefig(save_path, dpi=300, bbox_inches="tight")
        plt.close()

    print(f"Heatmaps saved in: {output_folder}")
```