|  |
| --- |
| **Algorithm 1**: Pseudocode for Breast Mass Detection |
| **Input** : image: Pre-processed image(height,width).  **Output** : Bounding boxes of detected objects.   1. SET predictions to EMPTY LIST 2. #split the image into grid of cells 3. cells=split\_image\_into\_grid(image) 4. #For each cell in the grid, predict bounding boxes and confidence scores. 5. **for** **each** cell in cells **do**: 6. prediction.append(predict\_bounding\_boxes\_and\_confidence\_scores(cell)) 7. #perform non-max suppression to remove overlapping bounding boxes 8. boxes,scores=perform\_non\_max\_suppression(predictions) 9. #Filter bounding boxes on confidence score 10. filtered\_boxes=[] 11. filtered\_scores=[] 12. for i in range(len(boxes)): 13. if scores[i]>0.5: 14. filtered\_boxes.append(boxes[i]) 15. filtered\_scores.append(scores[i]) 16. #return filtered boxes and scores 17. return filtered\_boxes,filtered\_scores |
|  |

**Input** :

image: Pre-processed image(height,width).

**Output** :

Bounding boxes of detected objects and scores of each bounding boxes.

1. SET predictions to EMPTY LIST
2. SET filtered\_boxes to EMPTY LIST
3. SET filtered\_scores to EMPTY LIST
4. #split the image into grid of cells
5. cells=split\_image\_into\_grid(image)
6. #For each cell in the grid, predict bounding boxes and confidence scores.
7. **for** **each** cell in cells **do**:

prediction.append(predict\_bounding\_boxes\_and\_confidence\_scores(cell))

**end**

#perform non-max suppression to remove overlapping bounding boxes

1. boxes,scores=perform\_non\_max\_suppression(predictions)
2. #Filter bounding boxes on confidence score
3. **for** i in len(boxes) **do**:

**if** scores[i]>0.5 **then**:

filtered\_boxes.append(boxes[i])

filtered\_scores.append(scores[i])

**end**

**end**

1. #return filtered boxes and scores
2. return filtered\_boxes,filtered\_scores

|  |
| --- |
| **Algorithm 1**: Pseudocode for Breast Mass Detection |
| **Input** :  image: pre-processed mammograph image of the breast.  **Output** :  Bounding boxes of detected objects and scores of each bounding boxes. |
| 1. predictions= [] 2. filtered\_boxes= [] 3. filtered\_scores= [] 4. cells=split\_image\_into\_grid(image) // split the image into grid of cells 5. **for** **each** cell in cells **do**: // For each cell in grid, predict bounding boxes & confidence scores.   prediction.append(predict\_bounding\_boxes\_and\_confidence\_scores(cell))  **end**   1. boxes,scores=non\_max\_suppression(predictions) //remove overlapping bounding boxes 2. **for** i in len(boxes) **do**: // Filter bounding boxes on confidence score   **if** scores[i]>0.5 **then**:  filtered\_boxes.append(boxes[i])  filtered\_scores.append(scores[i])  **end**  **end**   1. return filtered\_boxes,filtered\_scores |

|  |
| --- |
| **Algorithm 2**: Pseudocode for Mass Segmentation |
| **Input** :  image: pre-processed mammograph image of the breast.  **Output** :  segmentation\_mask: A binary image with 1 indicating the breast mass and 0 indicating the background.  object\_area: The area of the breast mass in pixels.  object\_percentage: The percentage of the image that the breast mass occupies. |
| 1. object\_area **←** 0 2. predictions = generate\_predictions(image) 3. segmentation\_mask = get\_segmentation\_mask(predictions)// get model's predictions. 4. **if** segmentation\_mask > 0.5 **then**: // Apply a threshold to the segmentation mask.   segmentation\_mask = 1   1. **else**:   segmentation\_mask = 0   1. **end** 2. **for** i in segmentation\_mask.shape[0] **do**: //row   **for** j in segmentation\_mask.shape[1] **do**: //column  **if** (segmentation\_mask[i, j] == 1) **then**:  object\_area += 1  **end**  **end**  **end**   1. object\_percentage = object\_area / image.size // Calculate percentage of image that breast mass occupies. 2. return segmentation\_mask, object\_area, object\_percentage |