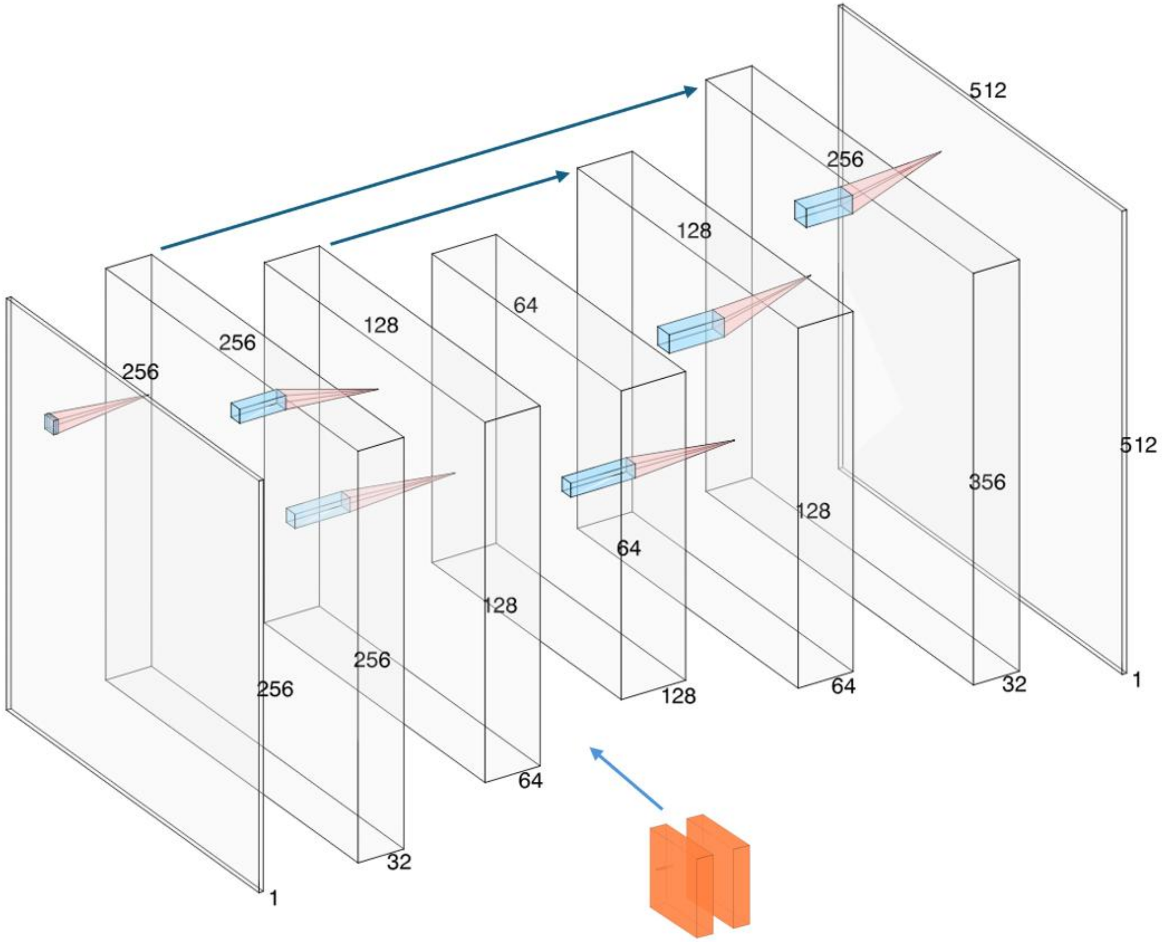
Andrew et al. Autoencoder (+ CBAM)

Adding a **CBAM attention module** didn't make the generated images better, as we can see by logging the **Channel** and **Spatial Attention Masks**:

```
Channel Attention Min-Max: 0.496 - 0.503
Spatial Attention Min-Max: 0.374 - 0.377
```

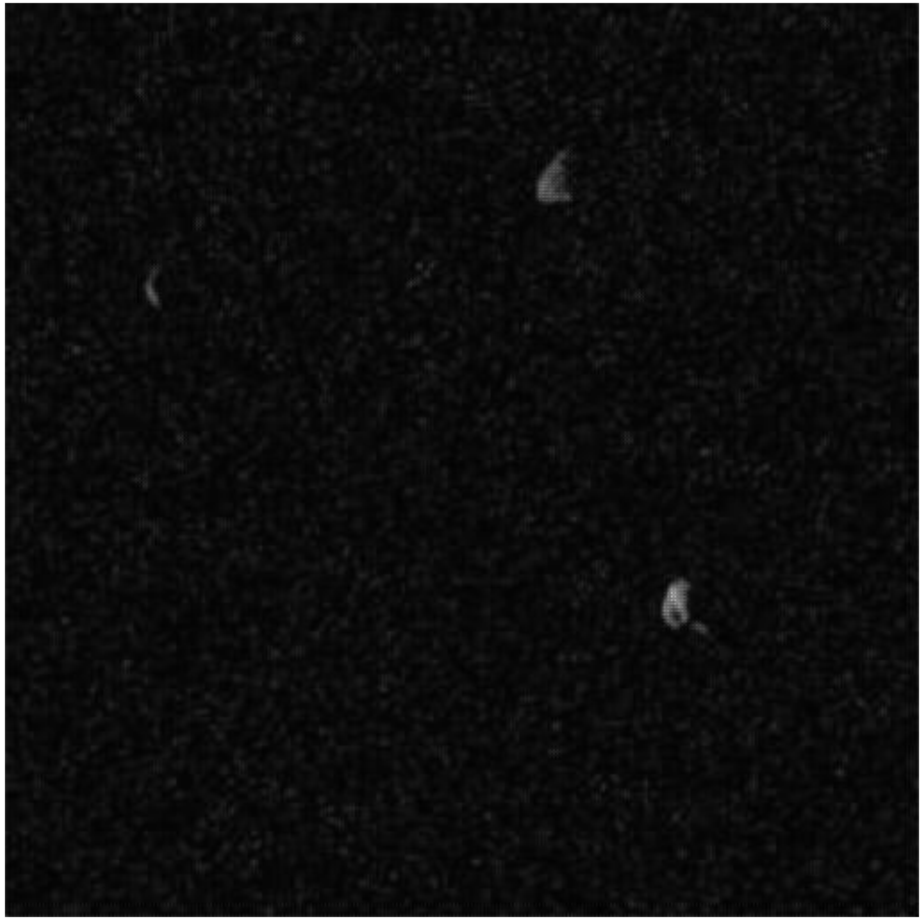
Which means that all the channel and attention **weights are very close to each other**, so there aren't any features the attention module favours over the others.

Spatial Attention mask from training image '*9022.jpg*'



SR results from the Autoencoder

Generated Image

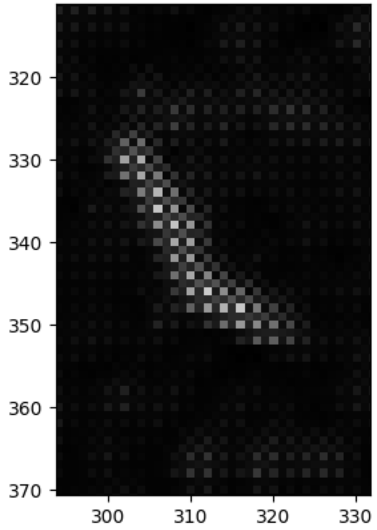


Hi-Res 9022.jpg

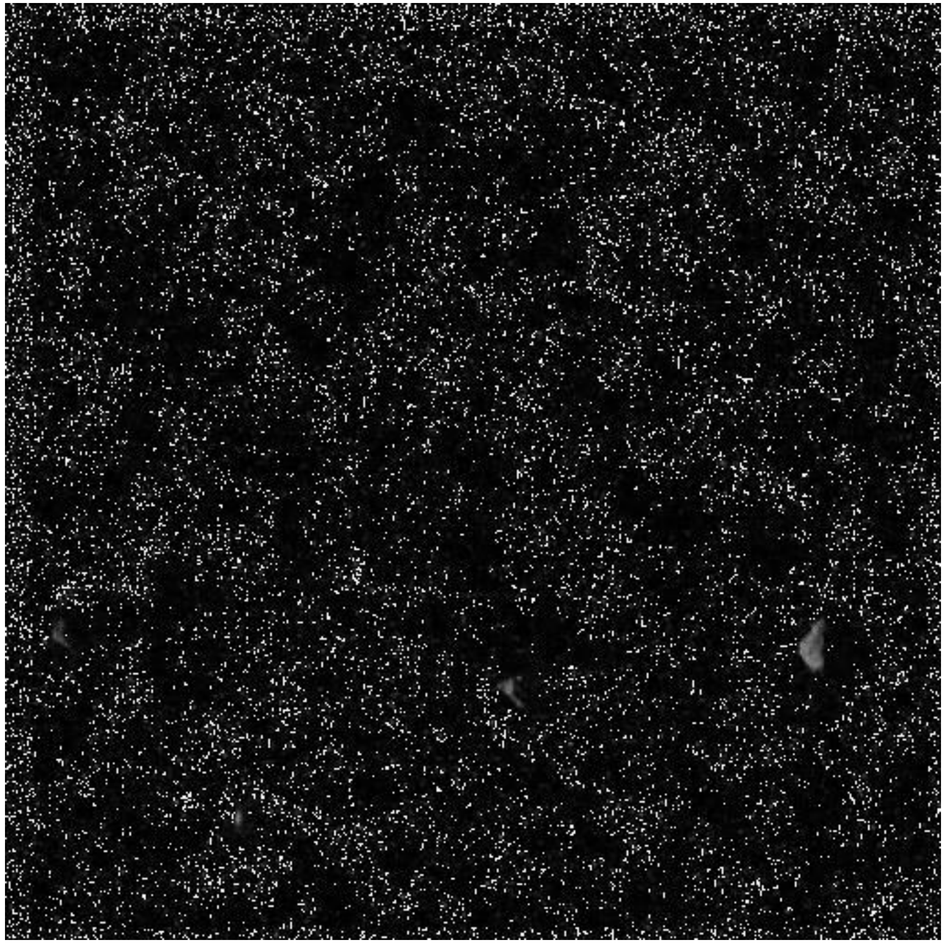
SR results from the Autoencoder

The model would produce a lot of **negative values in the generated image**, this would result in a very **noisy image**, since the negative pixels were displayed as pure white by the `to_pil_images()` method.

By **removing the negative values** with `clamp(0, 1)` or by using another display function, like matplotlib the noise would go away, but it would leave a **grid-like texture** all over the image.



Detail of one of the **debris after removing the negative values**



Noise generated by the **negative values**

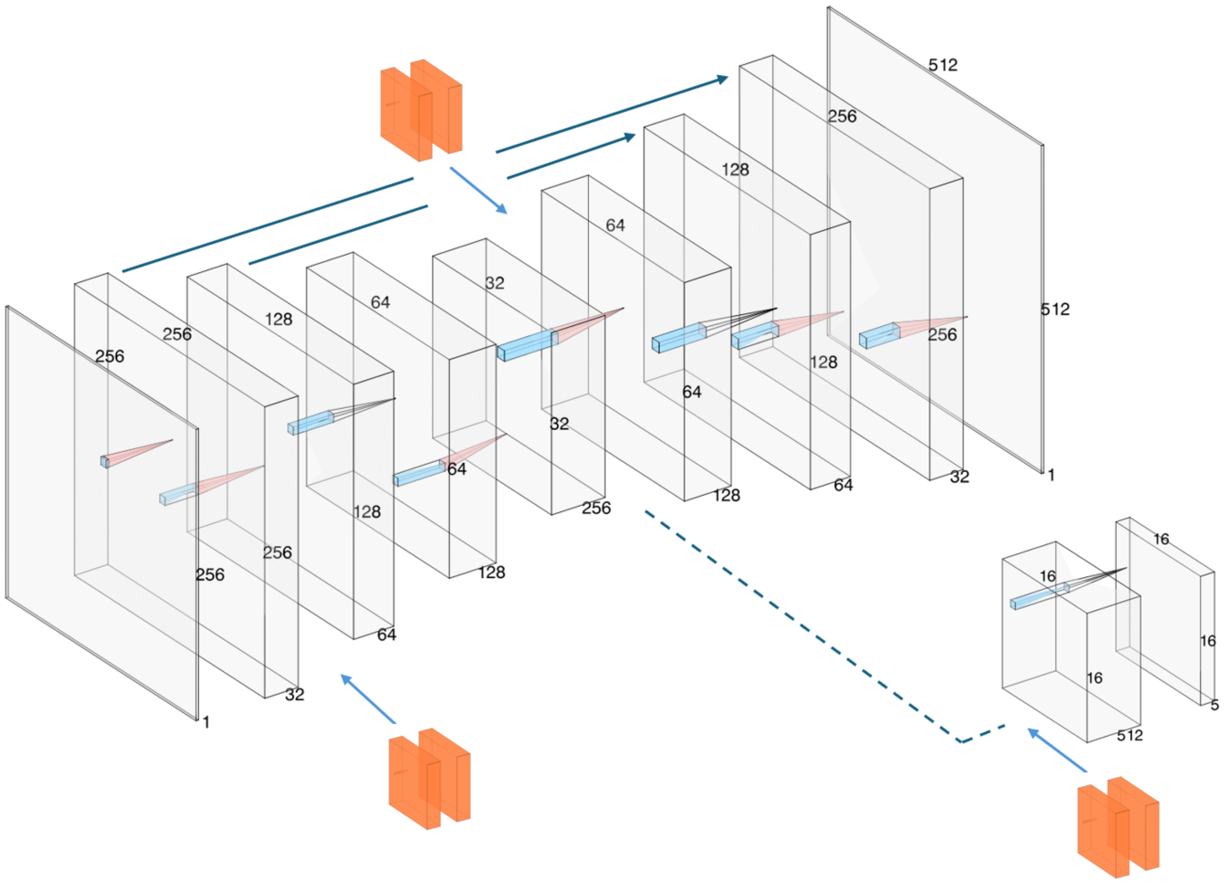
Autoencoder with YOLO detection and CBAM

To get better results from the images, and to simultaneously perform **Object Detection** I implemented some **modification** to the original Autoencoder model:

The **Object Detection module** is inspired by *Joseph Redmon's et al. 'You Only Look Once: Unified, Real-Time Object Detection'.*

The **deeper network** allows a better extraction of the image's features, resulting in a **more accurate reconstruction** of the high-definition image and the prediction of the **bounding boxes** for Object Detection.

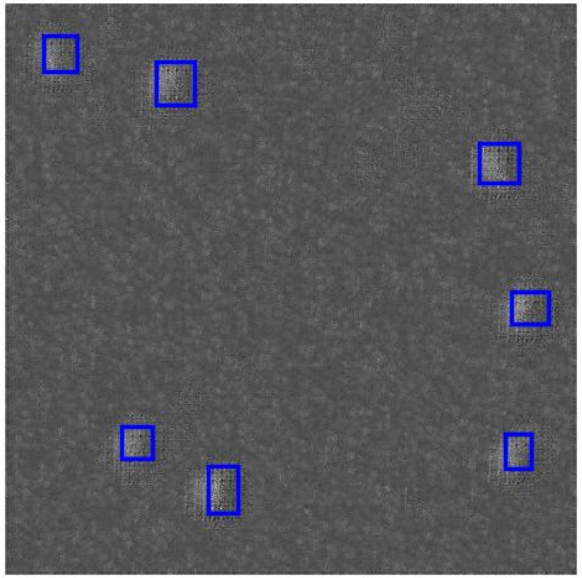
The **three CBAM modules** allow the model to focus on the **important parts** of the image needed **for each of the tasks**.



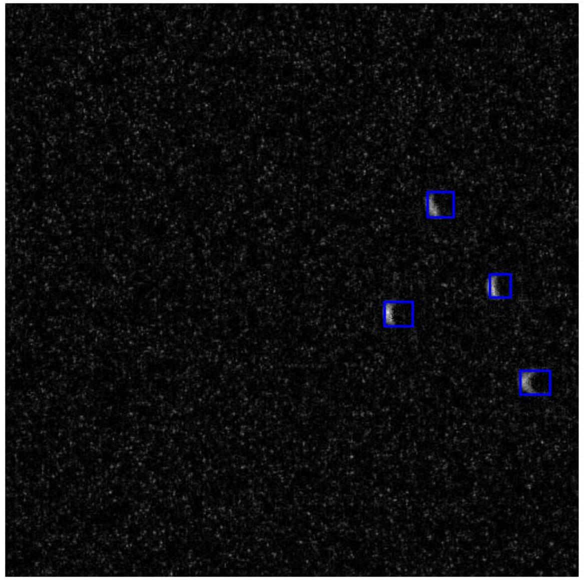
Training the YOLO-Autoencoder

To test the efficacy of the model I trained it for 20 epochs, using as loss function the sum of the Yolo loss for SR and the MSE loss for OD:

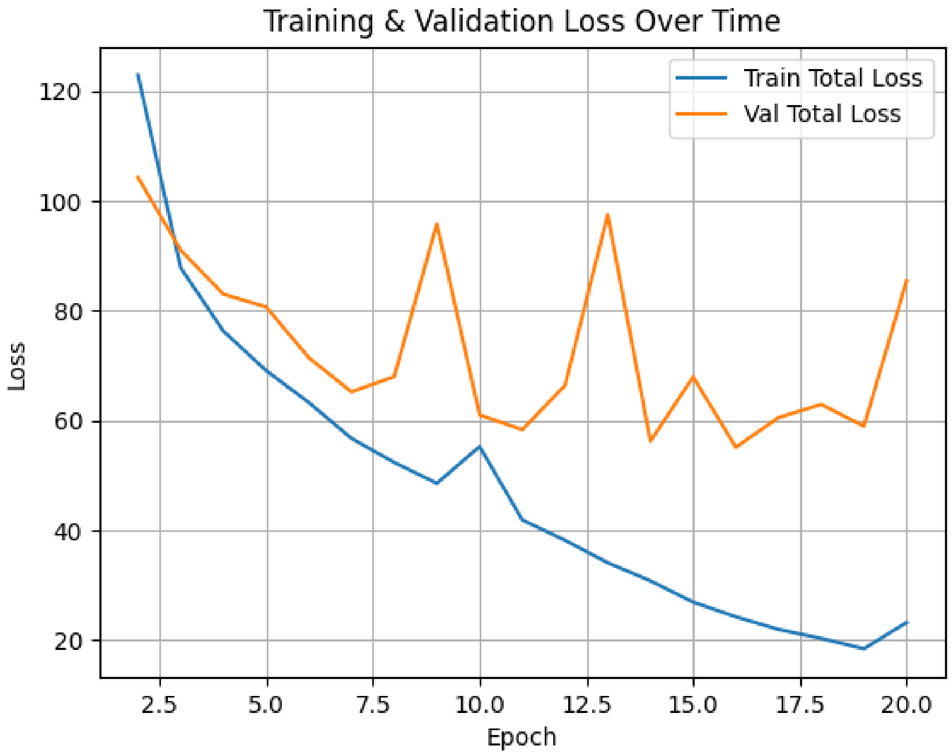
```
loss = YoloLoss() + lambdaSR * MseLoss()
```



Generated Image and bounding boxes from **Epoch 2**

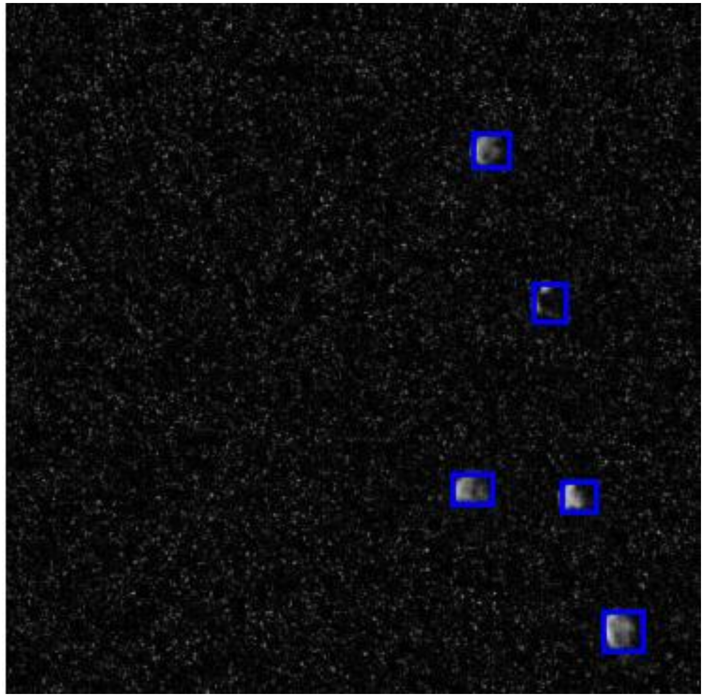


Generated Image and bounding boxes from **Epoch 6**

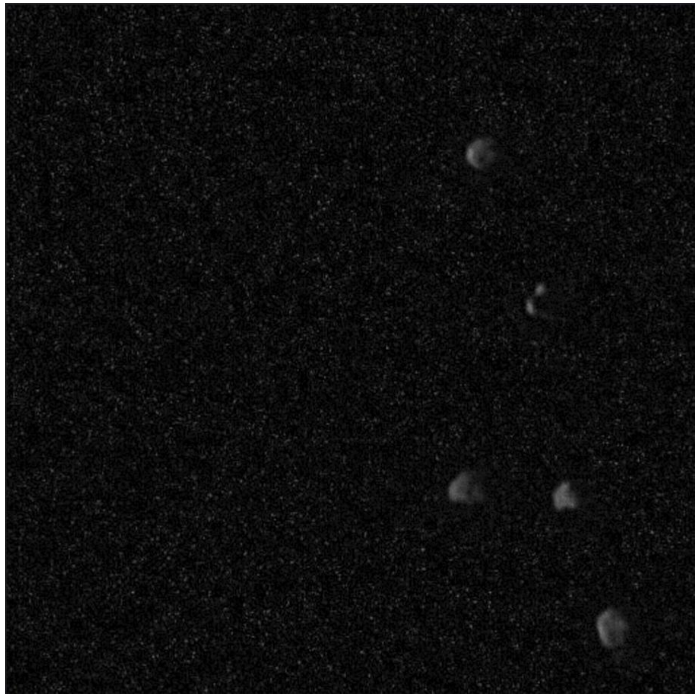


Training and Validation Loss over the Epochs

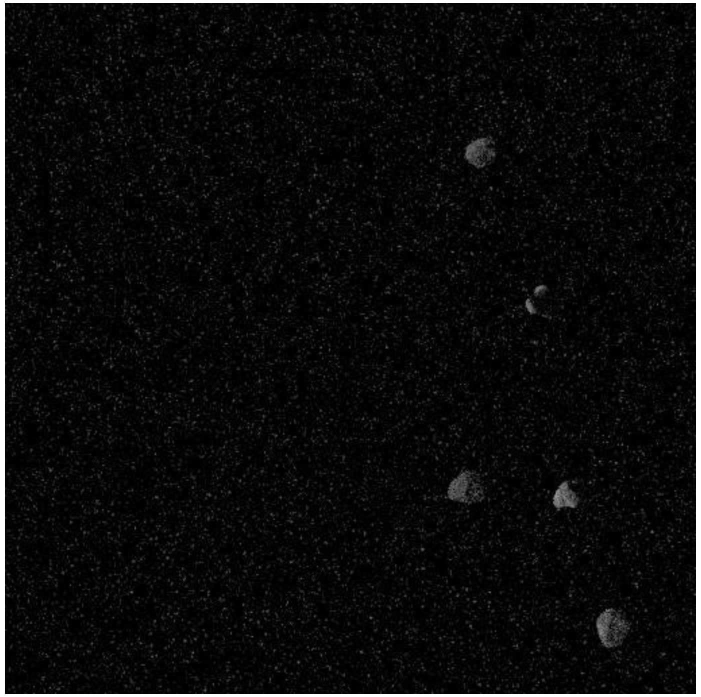
Results of the YOLO-Autoencoder



Reconstruction of test image ‘1998.jpg’ with predicted bBoxes



Reconstruction of test image ‘1998.jpg’



Test image ‘1998.jpg’

After only 20 epochs the model is **reliably identifying the debris** in the images, both in the images containing a couple of debris and in the images containing more.
While the model is **accurately reconstructing the background**, the **debris** themselves are **blurry**.

Results of the YOLO-Autoencoder – CBAM

Contrary to the original Autoencoder model, the **CBAM** modules are **selecting** some **relevant features** from the different layers of the network:

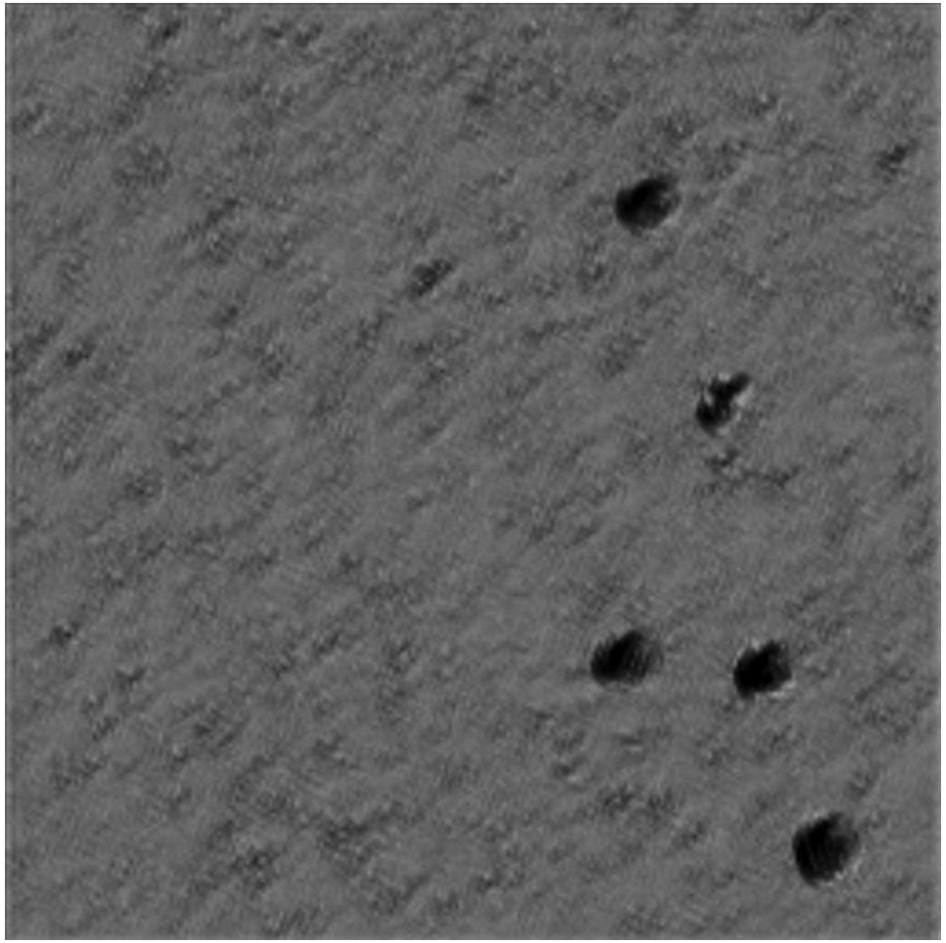
1st Module:
`Channel Attention Min-Max: 0.005 - 0.998`
`Spatial Attention Min-Max: ~0.00 - 0.609`

2nd Module (Detection):
`Channel Attention Min-Max: ~0.00 - 0.999`
`Spatial Attention Min-Max: 0.468 - 0.955`

3rd Module (SR):
`Channel Attention Min-Max: / - /`
`Spatial Attention Min-Max: / - /`

All the attention masks have values in a **large range**, so they are selecting different features.

Most notably, the **attention mask** of the first module is **sharp**, and it is picking out the debris.
The **assigned values**, though, are **opposite** to what we would expect:
The **debris are given the least importance** and the background the most!
This explains why in the reconstruction just the debris are blurry.



Spatial Attention mask from the **first CBAM** module
1998.jpg