PARTICLE SIZE DETECTION AND QUANTIFYING ITS DISTRIBUTION

Sneha Bhattacharjee (112001056)
Rapeti Siddhu Neehal (112001034)

CHALLENGES

there are a multitude of challenges that we have identified in the project.

majority of the smaller particles in the images are due to gaussian noise. the differentiation of this from actual small scilica particles.

SOLVED

rejecting the larger grooves from the image as noise. grooves caused due to improper polishing(sanding) etc

SOLVED

A good portion of the particles that are adjacent are broken chunks from the same parent particle, such chunks must be counted as different particles as once broken their properties differ.

SOLVED

automatically determining the scale from an image
currently have extracted the scale from the metadata of .tif files, but it is not universal and differs for different proprietary images, in our case only for ZEISS SEM images

PARTIALLY SOLVED

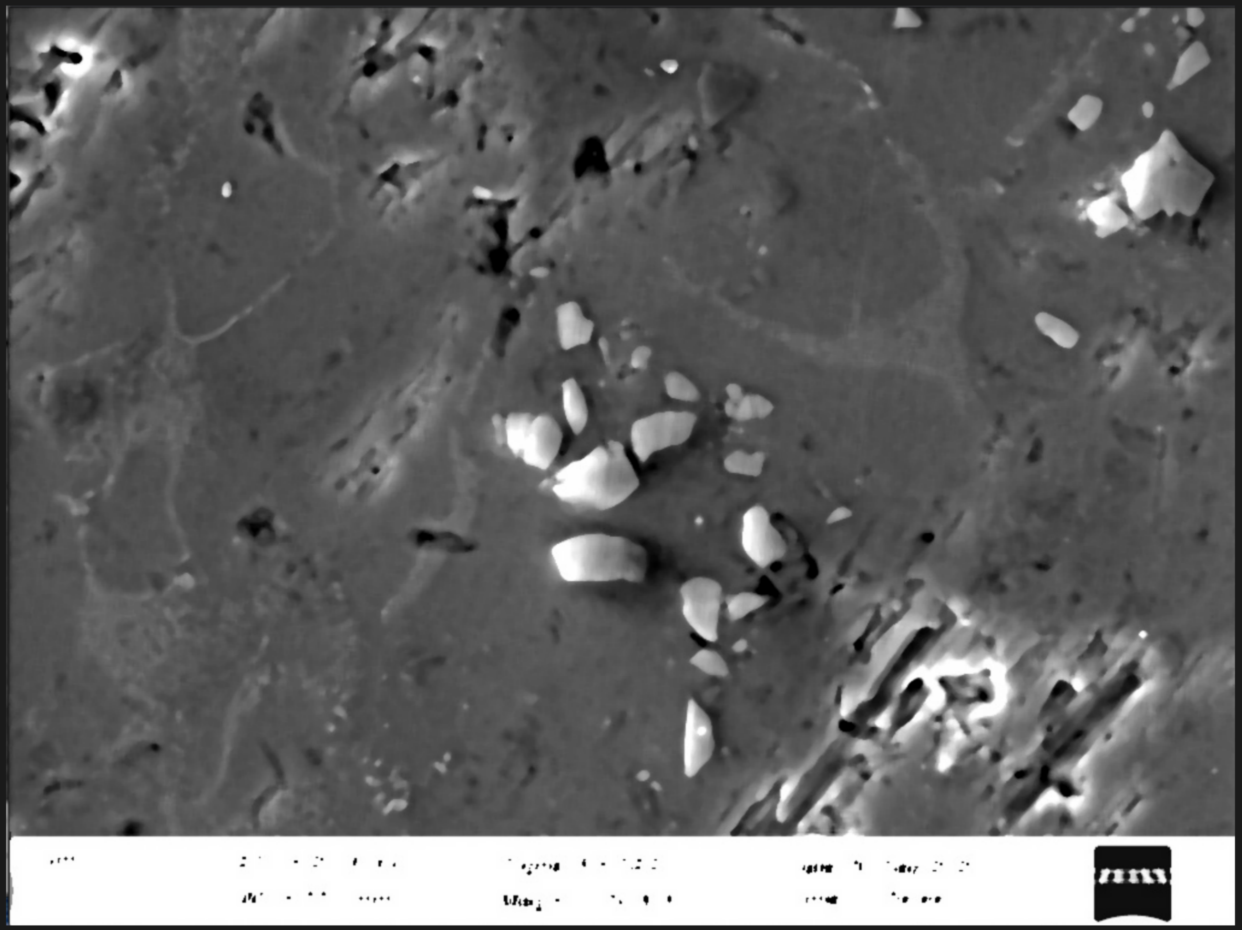
DENOISING

1. (most compatible) Non-local means denoising, it preserves certain textures. It was able to get rid of much of the Gaussian noise. It also doesn't excessively blur the smaller particles.
2. (best)Bilateral blur seems to be the better option as it helps reduce noise without having any drastic effect on the edges.
3. Median blur works in this use case as a good amount of gaussian noise.
4. B3MD manages to smoothen the matrix, but it reduces the colour contrast between the particles and the medium.
5. Total variation denoising: Much like nlm but the edges become less sharp and the image is smoother.

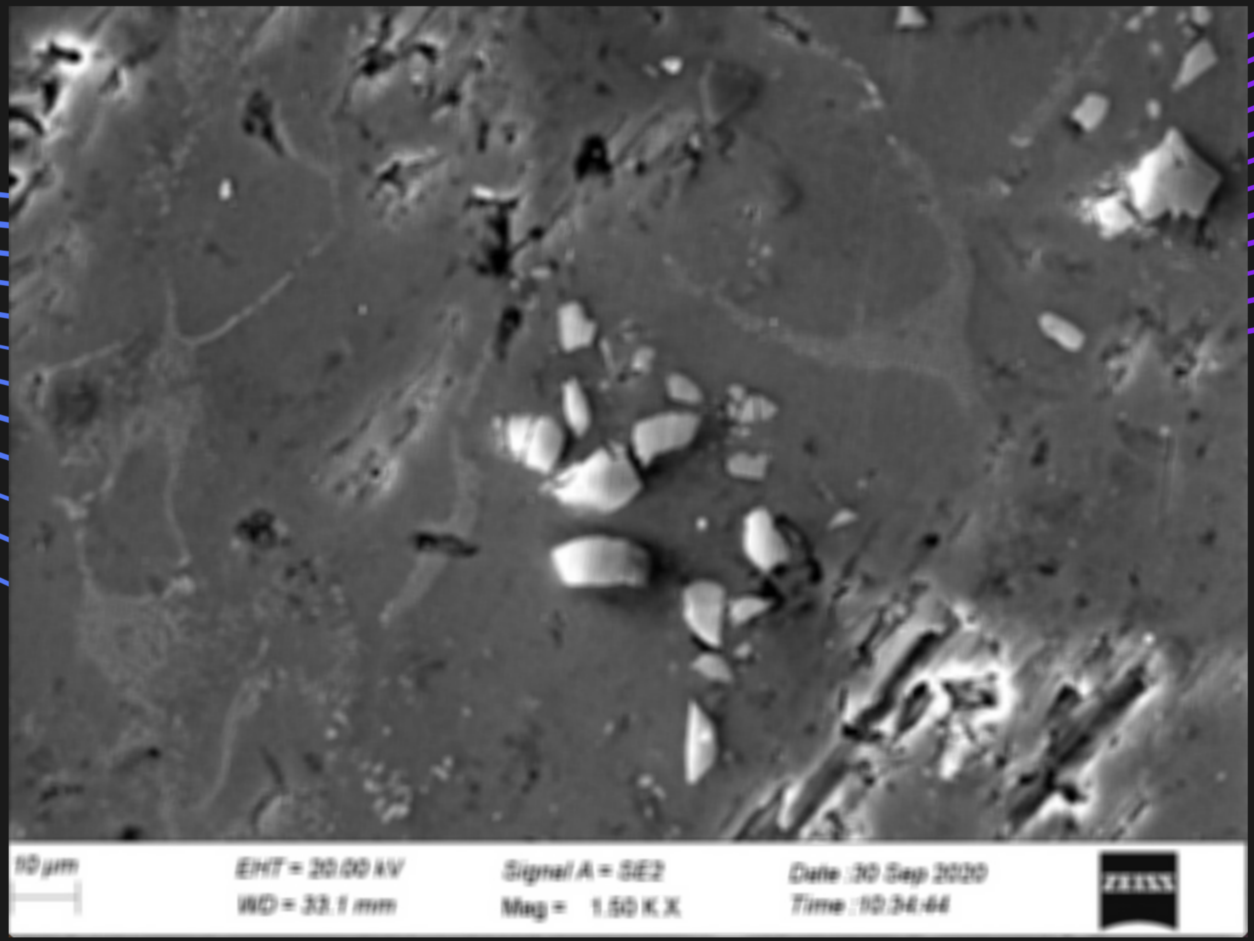
advanced and complex way: using morphological operations to remove noise (morph function) it uses series of erode() and dilate() function to reduce the noise while preserving the edges.

- **Note**: it is possible to apply multiple filters but then the scope of this project might get limited to only a particular type of images processing (silica alloy) as it might cause overfitting.

bilateral filter


median blur

B3MD

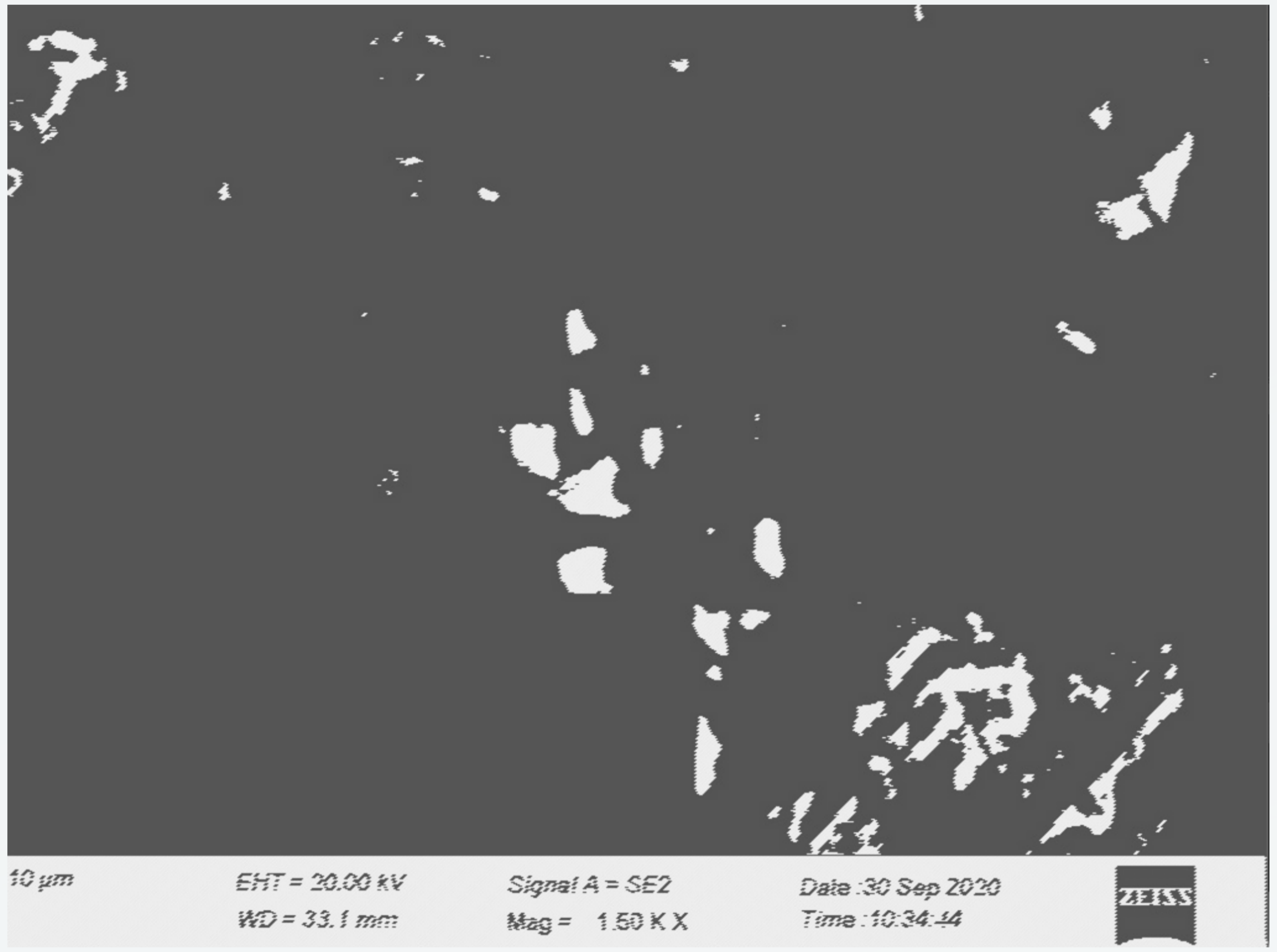

non-local means denoising

DIFFERENT TECHNIQUES

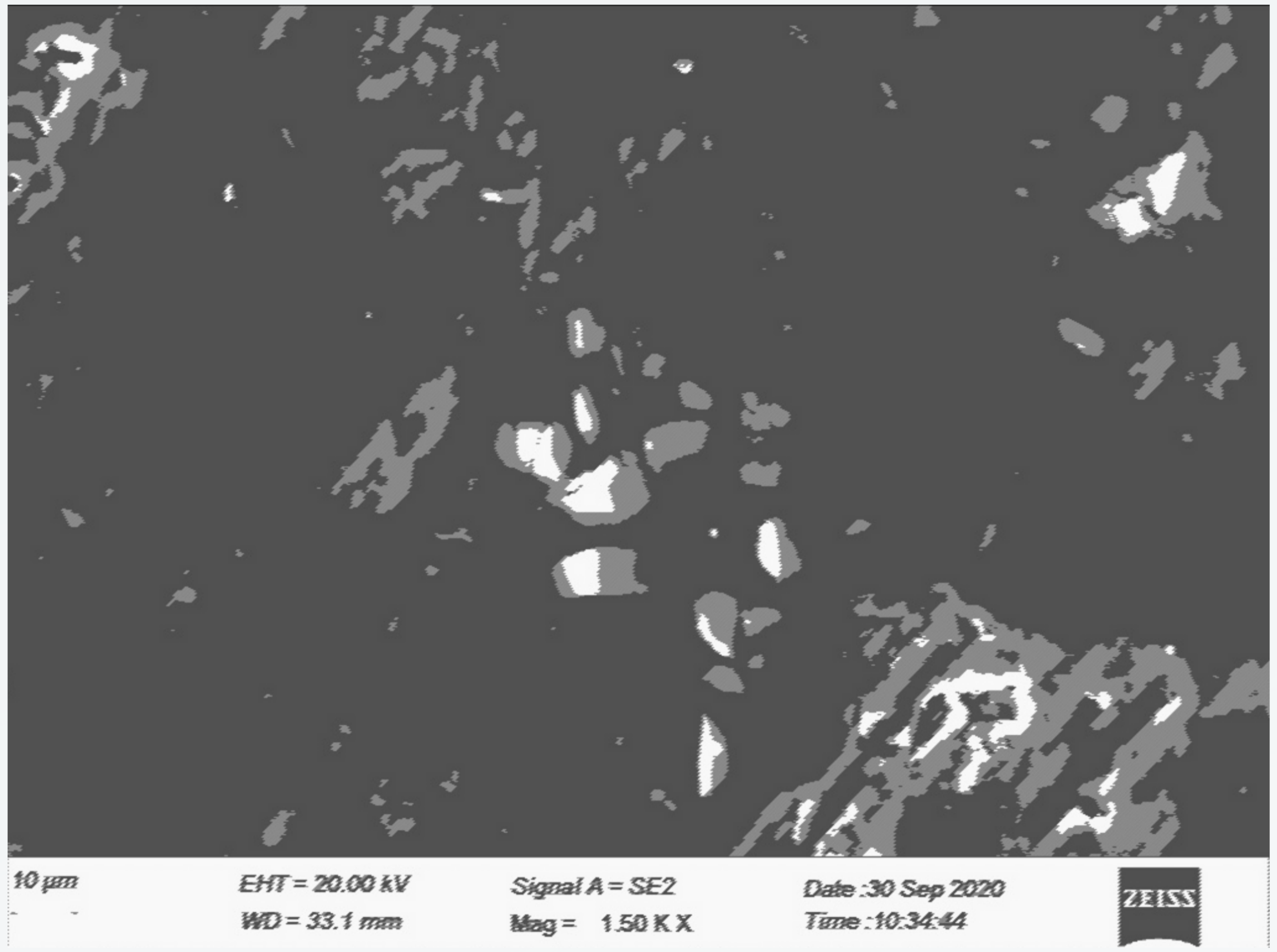
- Kmeans thresholding: we got best results with k = 3, where it clusters the pixels into 3 groups.

- Otsu's thresholding: finds the value that minimizes the intra-class variance of the foreground and background pixels in the image. the threshold value is chosen such that the intra-class variance is minimized.

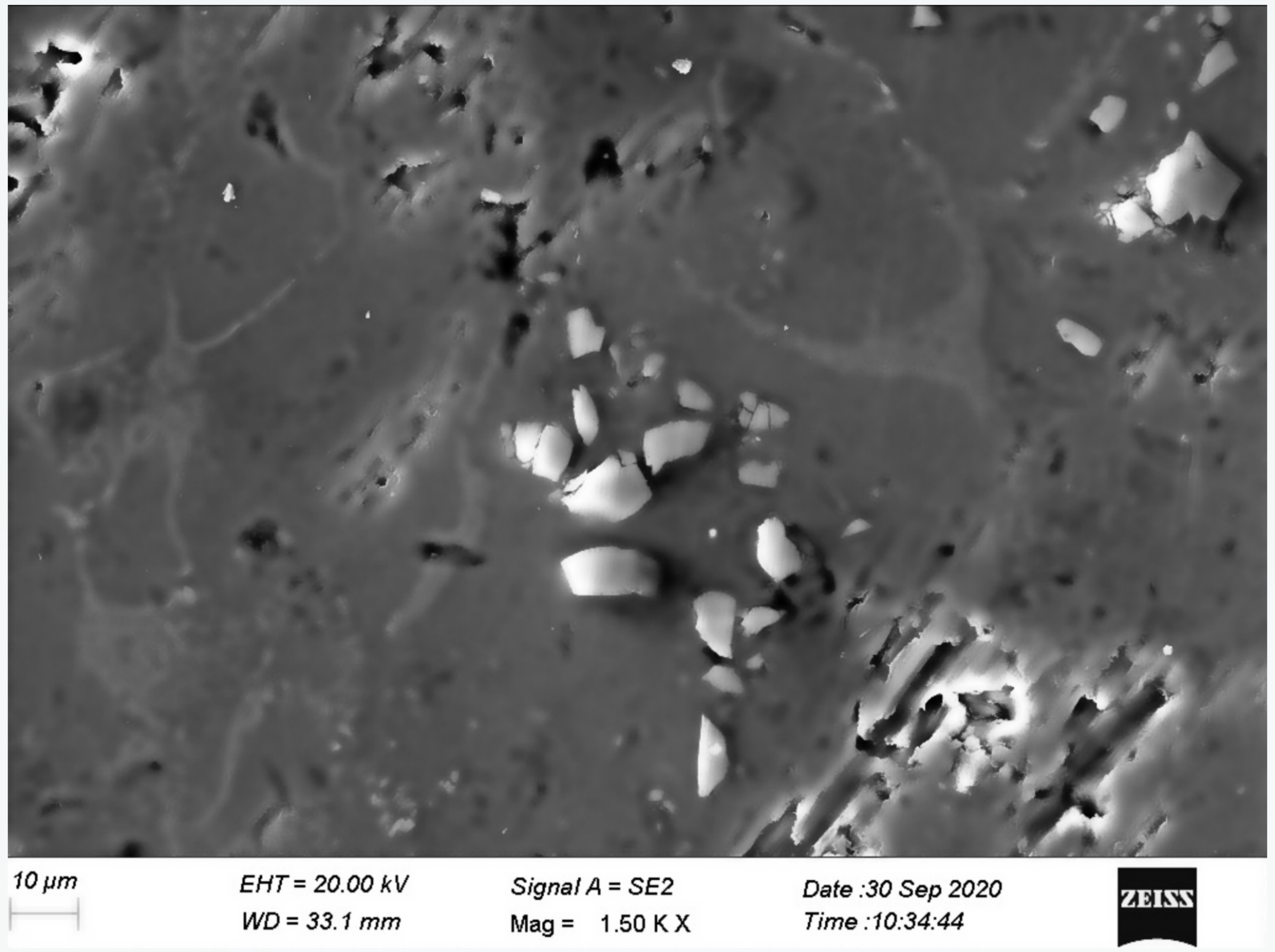
- Adaptive thresholding: theoretically this better fits our scenario as all the foreground is not very cleary differentiable and the silica particles (foreground) vary in the intensities.



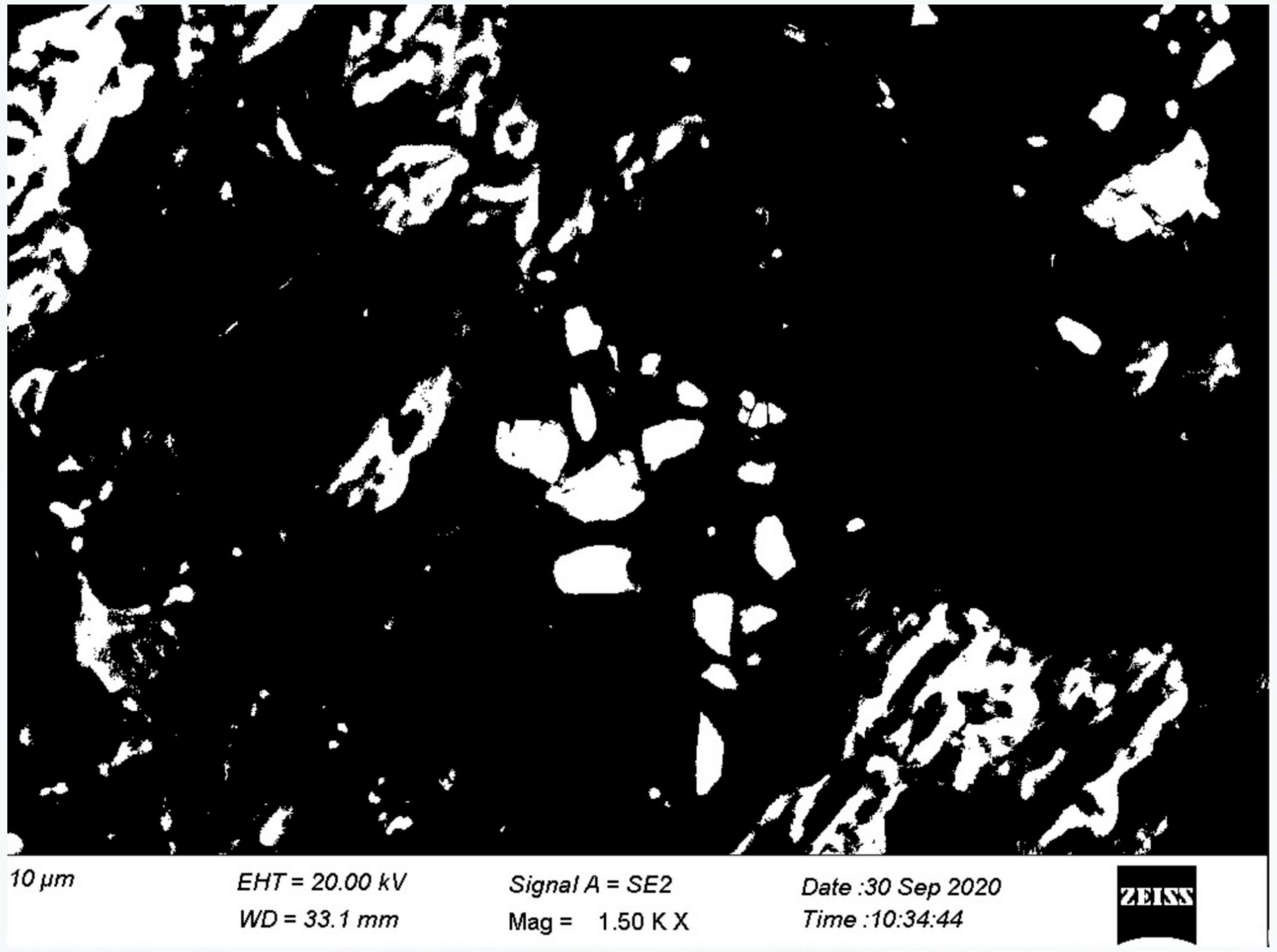
k = 2, kmeans thresholding



k = 3, kmeans thresholding



otsu thresholding



adaptive thresholding

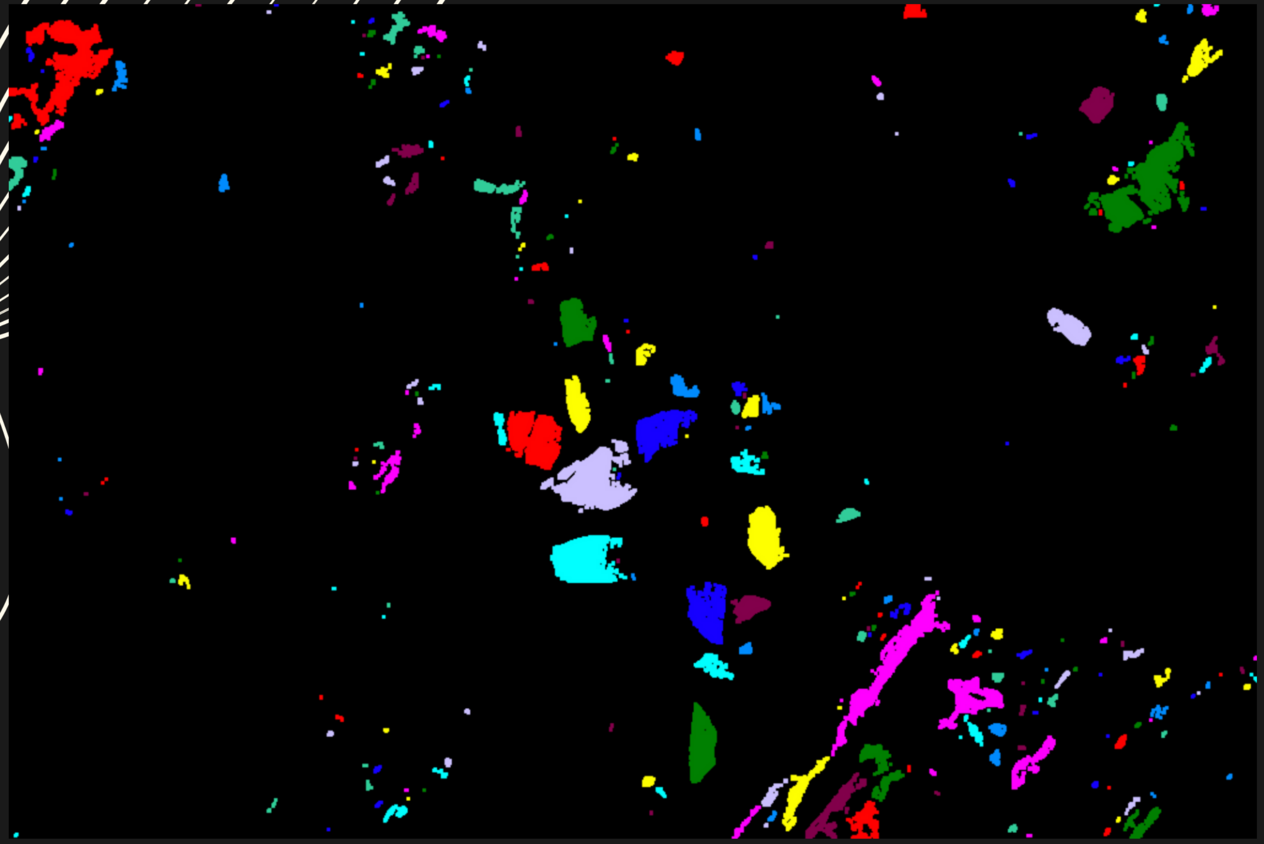
REMOVING GROOVES

- we used findContour() to detect the contours. these contours will be stored in a list and we used perimeter^2 / area of the contour as the metric to remove the grooves because the grooves will have this ration larger when compared to the silica particles as the particles tend to be closer to circle/square and not narrow like grooves.

- these grooves tend to have intensity similar to that of the particles which makes it difficult to remove them using contrasting methods.

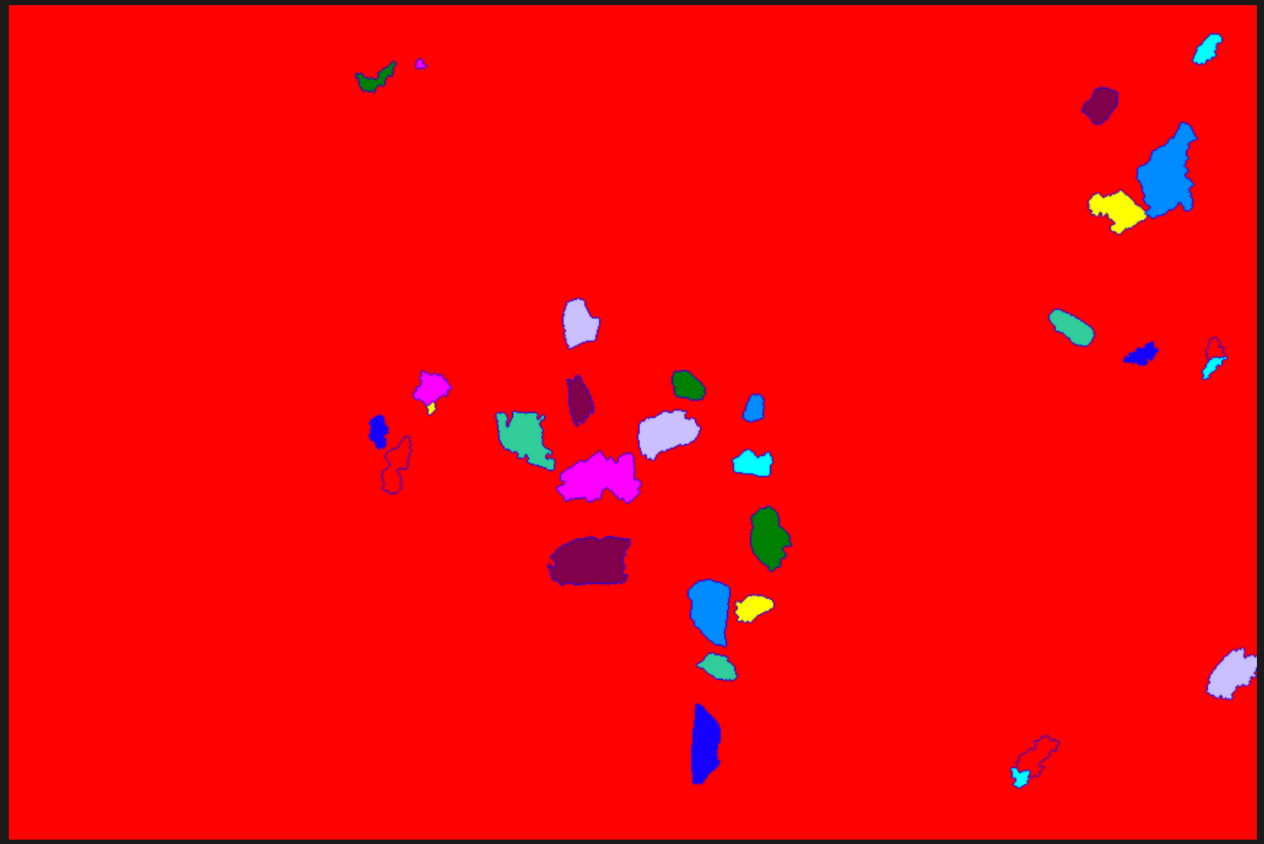
- some of the particles are silica particles are very close to each other to segment such particles we used watersheding.

- **THE MAGIC OF WATERSHED- it segments partially fragmented particles, most of the noise that was identified as particles get dismembered.**



contour detection

accepted particles



watershed segmentation

rejected groves

RANDOM FOREST

Using a random forest based model:

- Using an available tool like *apeer.com* to annotate the images to test how good the annotations are.
- Manually annotate images
    - Thresholding here is the most complex task since the shapes of particles of all sizes must be captured accurately
    - Hence we are training a model to take the images and their annotations as labels and figure out the annotations based on a set of features.
    - Features include intrinsic properties of the image, and various filters applied on them, Gaussian, Sobel, Gabor etc...

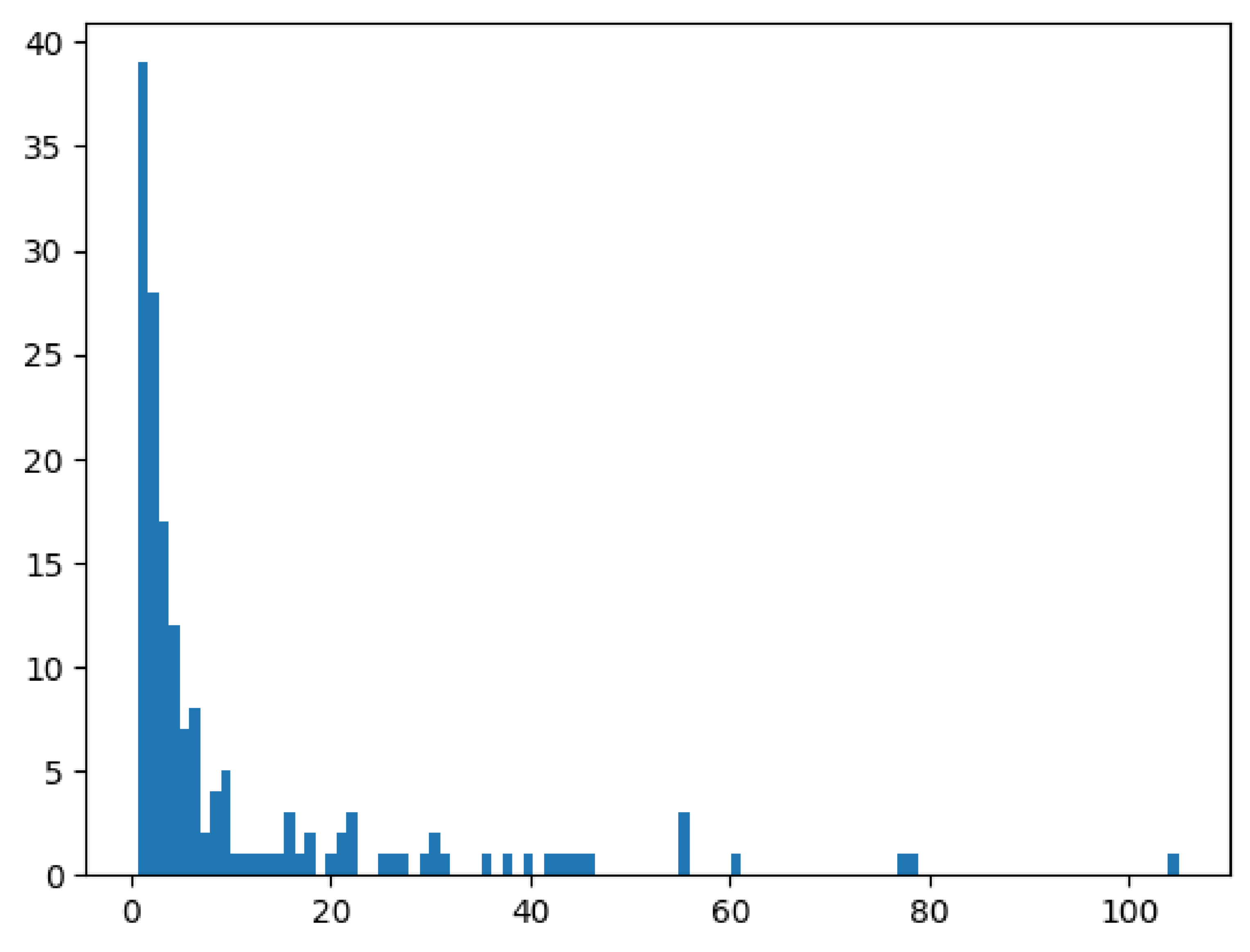
RESULTS

We have found/ plotted the following:

- Particle size distribution histogram
- Hopkins clustering metric
- Kolmogorov-Smirnov test

```
H = hopkins(centroids)
H
```

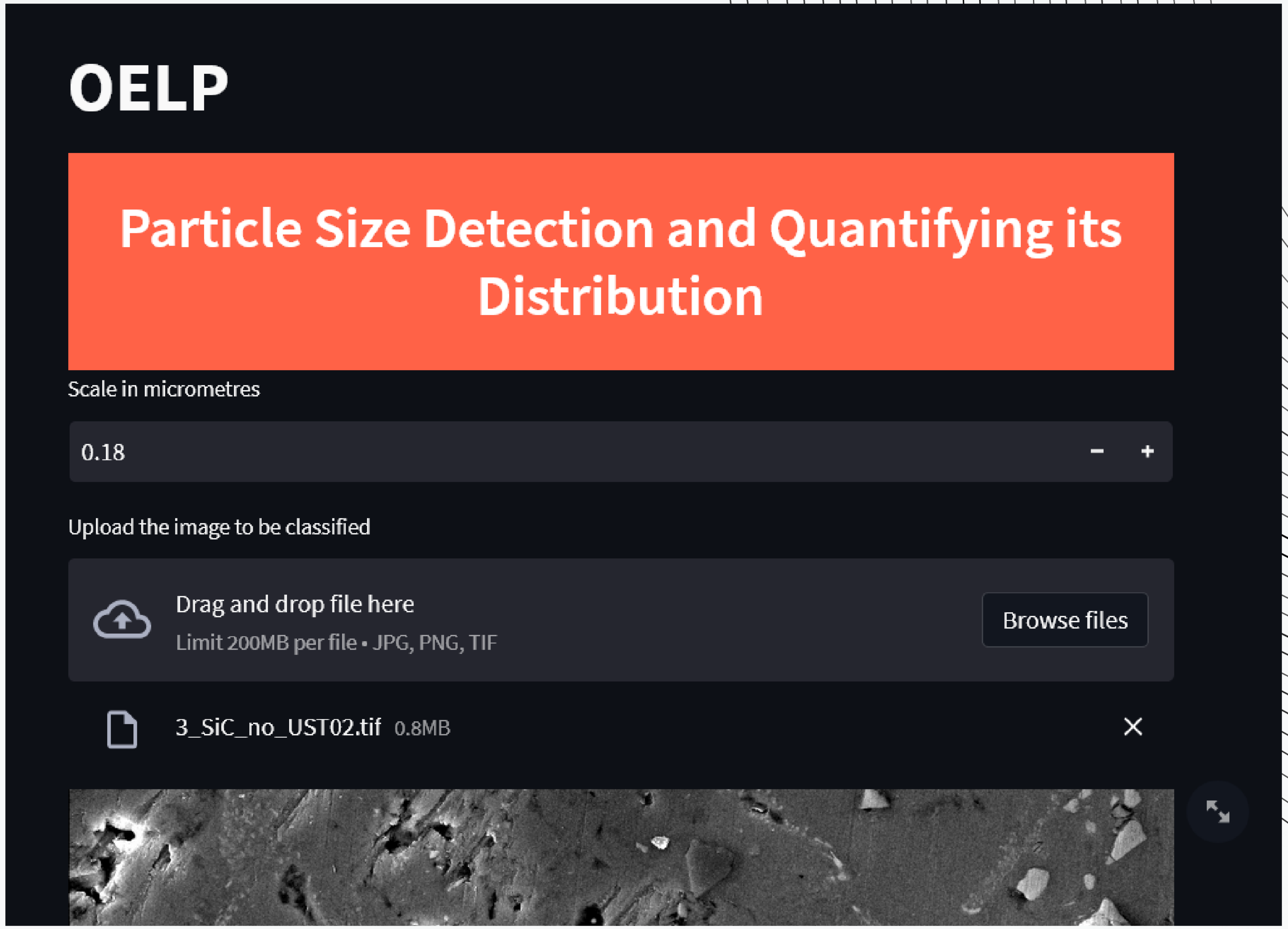
```
0.5062394333979032
```



DEPLOYMENT

We used streamlit to deploy the various steps

- The scale of the image is taken in as a user input
- The image can be in .tif. .jpg or .png formats
- It displays the histogram and the Hopkins metric



THANKS