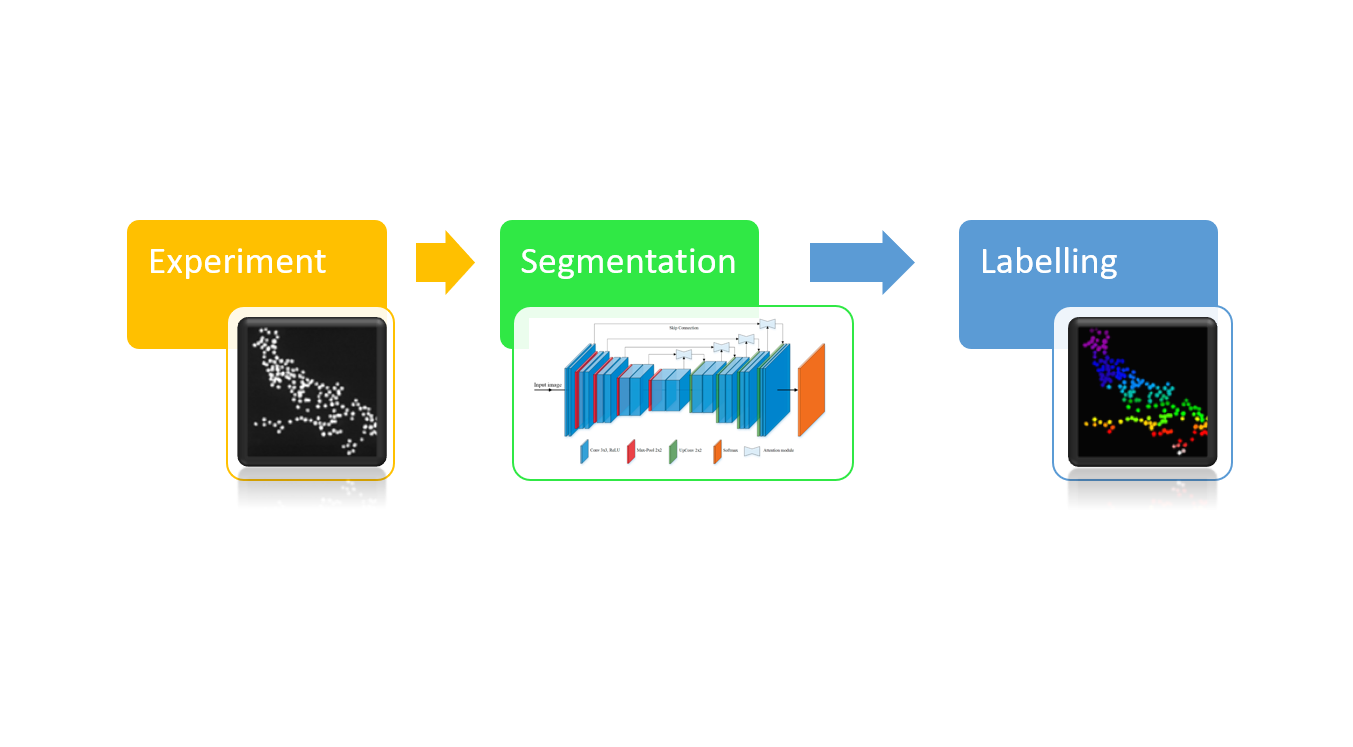
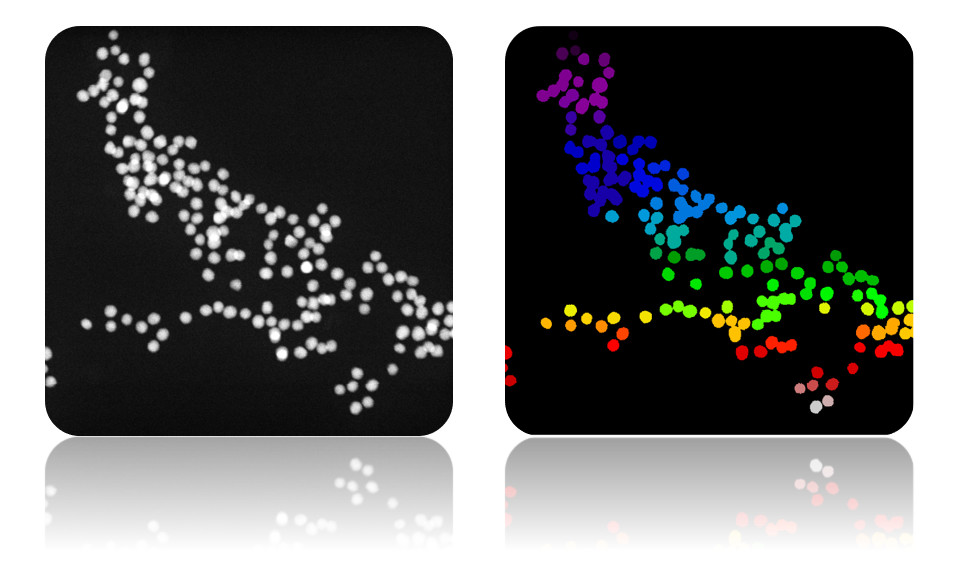
# Particle separation with clustering methods and auto-thresholding.

## Project description

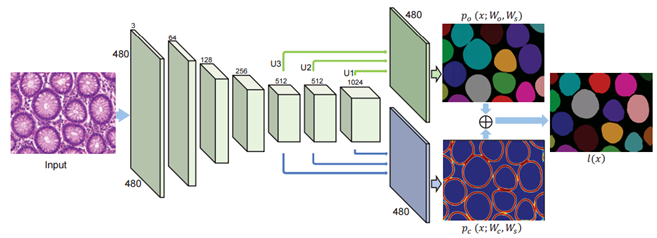
In this project I demonstrate an app that automatically thresholds particles and separates the thresholded masks into various colour coded entities. This type of separation is useful in cases where we have segmentation masks from a neural network segmentation that we want to further separate into various components. For demonstration we will use an example image of nanoparticles from a TEM image from <https://imagej.net/ParticleSizer>. The schematic below shows an example oh how something like this would work.



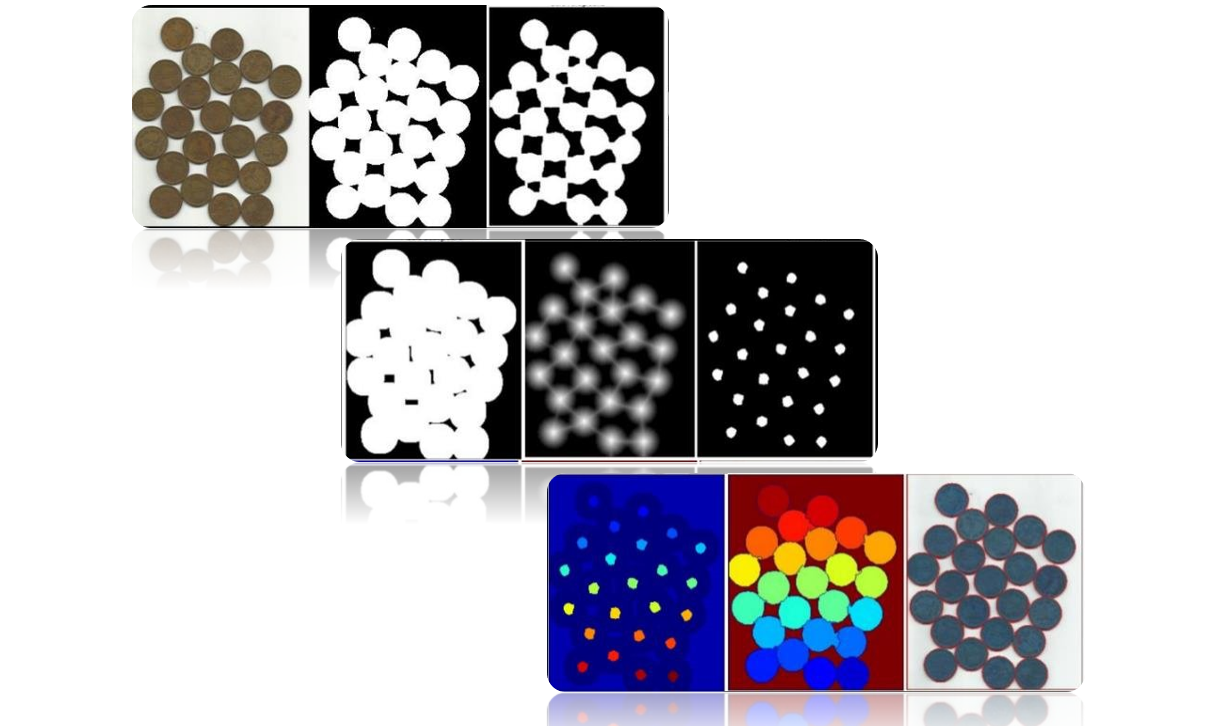
A closer look at the experimental data and labelled data shows that most of the particles have been clumped together into a single colour.



Modern deep learning approaches such as MASK-RCNN and this Nvidia blog post here <https://news.developer.nvidia.com/diagnosing-cancer-with-deep-learning-and-gpus/> showing how this approach can be used.



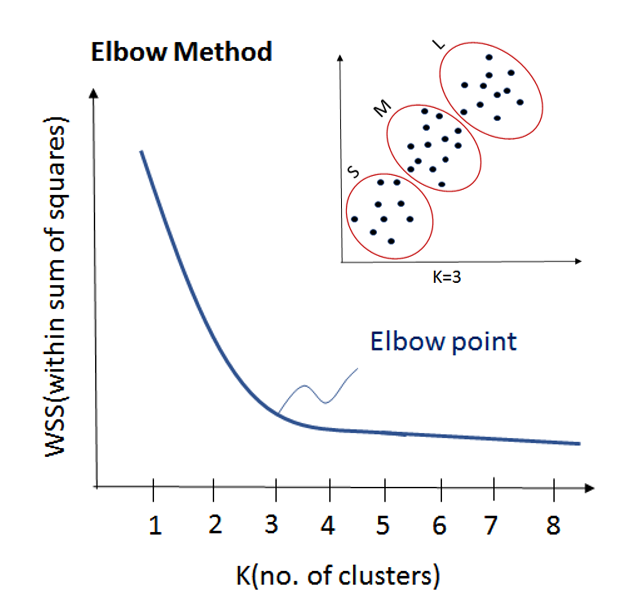
The approach shown above requires a few thousand high quality hand-labelled images. Often times these are not available for the vast number of samples analysed within a lab. Therefore traditionally, a “classical” image processing technique called watershed is used to separate the particles. <https://www.researchgate.net/publication/327552725_Brain_Tumor_Segmentation_Using_3D_Magnetic_Resonance_Imaging_Scans>



Whilst watershed is a well-known technique for doing this type of analysis, here we will explore something a bit different using k-means clustering.

## Segmentation method and clustering

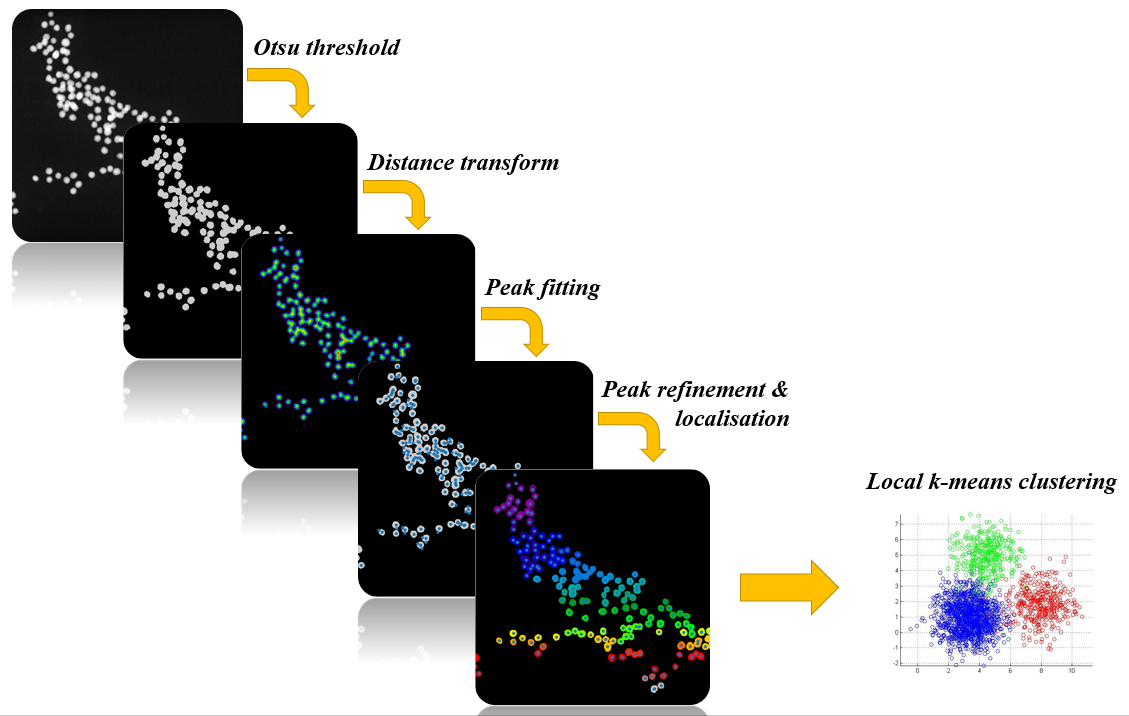
First, we need to segment the particles. As the particles are on a simple black background, we can use Otsu’s method <https://en.wikipedia.org/wiki/Otsu's_method> to identify the threshold mask. After thresholding, one can think of converting the images into a point cloud and using the elbow method shown below to calculate the number of optimal clusters.



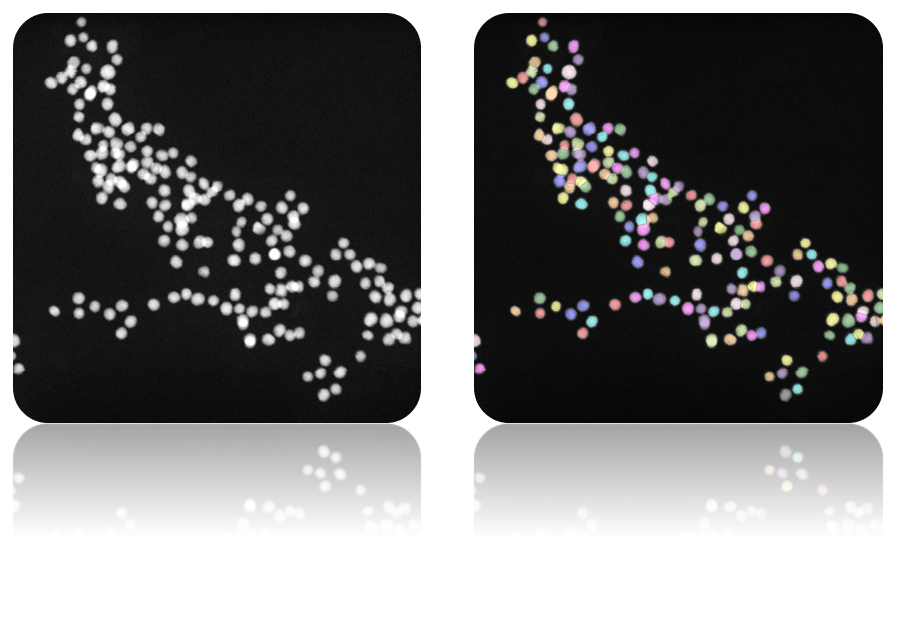
Credit : <https://www.edureka.co/blog/k-means-clustering-algorithm/>

The problem of doing this with images is that it is quite computationally expensive as the WSS metric must be calculated for every Kth step. Also, in real samples, it is difficult to determine where the elbow is.

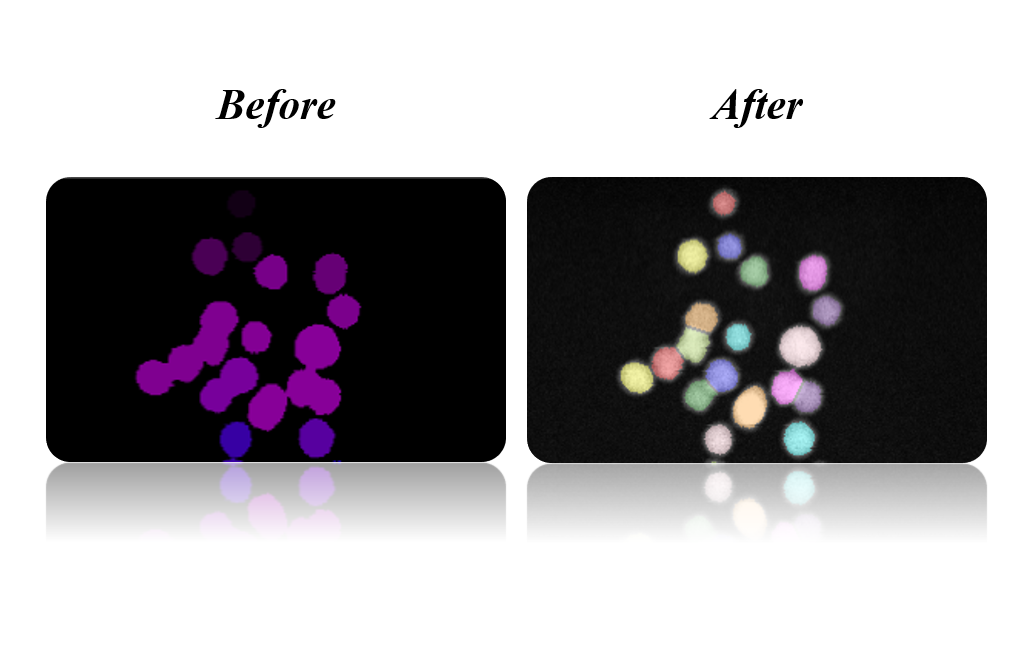
An alternative approach would be to apply a apply a distance transform over the thresholded mask and add some peak fitting to find the centres of the particles. The particle peak positions can be overlaid over a standard label mask. Then for each mask, a local k-means clustering can be performed based on the number of peaks present. This workflow is summarised below and it dramatically improves the fitting.



The resultant output of the labelling in shown below.



A close-up comparison of the before and after k-means separation for one of the regions is shown below.



## Demonstration of the workflow as a web application

The complete workflow described here can be packaged into a web application shown in the video below. In this demonstration the nanoparticles are separated from the background using Otsu’s method and the separation of particles within the mask is done using the k-means clustering method proposed.

[insert video here]

## Closing remarks and things to improve on.

Here we have seen an example of how a classical machine learning technique can be used on the output of a segmented mask from either a traditional thresholding method or a neural network. One thing that can be improved here is using a more sophisticated clustering algorithm such as Spectral or Gaussian Mixtures. However as most of the particles in this situation are round, k-means works just as well. Other things to consider are implementing a GPU accelerated version of this using RAPIDS or a TensorFlow backend.