Using a Self Organising Map for Clustering of Atomistic Samples

Aquistapace F.

Facultad de Ciencias Exactas y Naturales, UNCuyo

November 16, 2021

Introduction

This software implements a Self Organising Map (also known as a Kohonen Network) for clustering of atomistic samples through unsupervised learning. It is written in Python 3.8.8 and depends on the external packages NumPy (version 1.20.1) [1] and Pandas (version 1.2.4) [2]. The core of the algorithm is a neural network composed of an input layer and an output layer, where the number of output neurons determines the number of groups the atoms are going to be classified into. This software is based on the work by J. Troncoso on applying a SOM for cluster analysis and lattice defects detection [3].

V. 1.0

Description:

The first version of this software implements a command-line interface through which the user can select the input file, choose which features of the data to use and set the parameters of the algorithm. This interface also let's the user know the actual status of the program, and the time elapsed when it is finished.

An immediate problem of this version is the normalization method in the pre-processing stage of the analysis. For every data column C, associated with a selected feature, the normalization is given by Eq. 1:

$$C_{norm} = \frac{C}{C_{max}} \tag{1}$$

Where C_{norm} is the normalized data column and C_{max} is the maximum value of the original column. This is not the ideal normalization method, since the range of the new column is now $[C_{min}/C_{max}, 1]$ (where C_{min} is the minimum value of C), instead of being $[0, 1]$. This issue is going to be fixed in the next version of the software.

The elapsed time shown when the algorithm finalises takes into account the training of the SOM, the classification process of the atoms in the sample and the writing of the output file. This last step is extremely inefficient and should be optimized in future updates of the software.

Testing:

BCC Fe Bulk With a Void:

As a simple test, a 25×10^4 atoms BCC Fe bulk with a void was analyzed. The sample was created and relaxed with LAMMPS [4]. Since the sample has no defects, the goal was to use the SOM to identify the void in the center of the simulation box. This was achieved using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$, clustering the atoms into 3 groups and using the centro-symmetry parameter and the coordination (via Coordination Analysis with $r_c = 6$), both calculated with LAMMPS, as features.

The results are shown in Fig. 1 and Fig. 3. Although the algorithm classified the atoms into 3 groups, no atom was assigned to group 1. For the previously specified parameters, the SOM didn't produce useful

results when clustering the atoms into only 2 groups. For V.1 of the algorithm, this analysis took around $26s$ to complete. The same sample is presented in Fig. 2 and Fig. 4, with the atoms color coded by the centro-symmetry parameter, for comparison.

HEA Nano-foam Under Compression:

Fig. 5 shows the result of applying the SOM to a compressed HEA nano-foam sample, with the goal of classifying the atoms into 2 groups. The network was trained using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$. The features used in this occasion were:

- Coordination, via Coordination Analysis with cutoff radius $r_c = 6$
- Centro-symmetry parameter
- Atomic volume, via Voronoi Analysis

On the other hand, Fig. 6 shows the same sample, with the atoms classified by structure type (FCC, HCP, BCC and Other) using the PTM algorithm. It can be observed that the yellow atoms in Fig. 5 correspond to the HCP, BCC and Other atoms in Fig. 6. While the atoms in the remaining group of the SOM classification correspond to the FCC atoms of the sample. Fig. 7 shows the atoms classified into group 1 (yellow atoms) in a slice of the sample and Fig. 8 shows all non-FCC structure types in that same slice of the sample. In this sense, the SOM is correctly identifying the atoms associated with defects and/or that belong to the surface of the sample. This analysis took $198s$ to complete.

A problem that has been noticed while performing this test is that the order in which the features are specified seems to affect the resulting classification of the atoms. In this case, the algorithm produced the results shown in Fig 5 and Fig. 7 when the features were specified in the order *coordination* $\rightarrow$ *centro-symmetry* $\rightarrow$ *atomic volume*, but couldn't get the same results when the order of the features was *coordination* $\rightarrow$ *atomic volume* $\rightarrow$ *centro-symmetry*. The origin of this issue is unknown, but this could be solved in future updates of the algorithm.

HEA Nano-foam Under Tension:

Fig. 9 shows the result of applying the SOM to a tensioned HEA nano-foam sample, with the goal of classifying the atoms into 6 groups so as to compare the results with a different unsupervised learning method, performed by N. Amigo with the K-means software he designed [5]. The SOM network was trained using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$, and the features used were:

- Potential energy (per-atom)
- Centro-symmetry parameter (12 and 18 neighbors)
- Structure type

- Voronoi coordination number
- Radial function coordination

On the other hand, Fig. 10 shows the same sample, with the atoms classified into 6 groups by N. Amigo's K-means clustering method, which also relies on per-atom cuantities. It can be seen that the SOM didn't perform as expected in this occasion, since it couldn't identify the slip planes present in some of the sample's filaments, which were correctly identified by the K-means algorithm.

A second test produced the desired results, as shown in Fig. 11, where the slip planes identified by the K-means method are now correctly identified by the SOM. In this case, the parameters used were the same as in the previous test, but the features used were:

- Atomic volume
- Radial function coordination
- Centro-symmetry parameter (12 and 18 neighbors)

Both tests took around $260s$ to complete.

Sumary

A summary of all the tests performed for V. 1 of the software is presented in Table 1. The same parameters were used on every test:

- $f = 1$
- $\sigma = 1$
- $\eta = 0.5$

V.1.1

Description

This version of the software addresses some of the major issues present in V.1.0. For every data column C, associated with a selected feature, the normalization is now given by Eq. 2:

$$C_{norm} = \frac{C - C_{min}}{C_{max}} \tag{2}$$

Where C_{norm} is the normalized data column and C_{min}, C_{max} are the minimum and maximum values, respectively, of the original column. This normalization method generates a column with range $[0, 1]$, solving the issue caused by Eq. 1.

Test	Nº of atoms	Features	N	Time elapsed	Results
BCC Fe with void	$\approx 25 \times 10^4$	Centro-symmetry g_r coordination	3	$26s$	Interpretable
Compressed HEA Nanofoam	$\approx 20 \times 10^5$	g_r coordination Centro-symmetry Atomic volume	2	$198s$	Interpretable
Tensioned HEA Nanofoam	$\approx 20 \times 10^5$	Potential energy Centro-symmetry (12 and 18 neighbors) Structure type Voronoi coordination g_r coordination	6	$260s$	Not interpretable
Tensioned HEA Nanofoam	$\approx 20 \times 10^5$	Atomic volume g_r coordination Centro-symmetry (12 and 18 neighbors)	6	$258s$	Interpretable

Table 1: Size of the sample, features, number of groups (**N**), performance and result of V.1 of the algorithm for every test.

The update equation for the SOM weights was also improved. In V.1.0, the training of the algorithm was regulated by Eq. 3:

$$w_{mn}^t = w_{mn}^t + \eta(t)h_m(t)\sum_{i=1}^{m}(x_n - w_{in}^t) \tag{3}$$

Where w_{mn}^t is the n-th component of the weight of the m-th output neuron at iteration t. Also, $\eta(t)$ is the learning rate and $h(t)$ is the neighbourhood function, both expressed as functions of the iteration number t. The error was caused by a mistake in the matrix representation of the equation and is corrected in Eq. 4:

$$w_{mn}^t = w_{mn}^t + \eta(t)h_m(t)(x_n - w_{mn}^t) \tag{4}$$

This corrected version of the update equation is the same that Troncoso used in his implementation of the SOM [3].

The slow writing speed of the output files is an issue that remains to be fixed.

V.1.2

Check the description of V.2.1 for a full breakdown of the updates, since these two version were developed in parallel.

V.1.3

Check the description of V.2.2 for a full breakdown of the updates, since these two version were developed in parallel.

V.1.4

Check the description of V.2.3 for a full breakdown of the updates, since these two version were developed in parallel.

V.2.0

Important: *This version was built from V.1.0 and developed in parallel with V.1.1. So all of the issues solved in V.1.1 are also solved for this version of the software.*

Description:

In this version, the program allows the user to analyse several input files sequentially after training the SOM with a user-specified training file. This can be used to get a consistent classification of the atoms of a sample at multiple steps of a molecular dynamics (MD) simulation. Other use for V2 is to test whether transfer learning is feasible, or not, for different samples.

The most significant change with respect to V.1 is that V.2 takes a python dictionary as the input for the program, which contains the parameters and file names necessary to run the software, instead of having the user specify them through the interactive command-line interface.

Testing:

HEA Nano-foam Under Tension:

The same HEA nano-foam under tension from the test performed for V.1.0 was used. In this case, two different dump files from said simulation were analyzed using one of them as the training sample. The atoms on each file were classified into 6 groups. Furthermore, the SOM network was trained using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$, and the features used were:

- Radial function coordination
- Centro-symmetry parameter (12 and 18 neighbors)

Fig. 12 shows the results of this analysis. It can be observed that, even though the SOM was trained with the sample at a strain of 50.5%, it correctly identifies the slip planes when the sample is at a strain of 55.5% without additional training. The classification done on the sample at a strain of 50.5% differs from the results obtained with V.1.0 of the algorithm for the same sample. In this occasion one of the available groups wasn't used, i.e. no atom was assigned to said group.

On a separate issue, Fig. 13 shows a comparison between one of the groups identified by the SOM and the atoms associated with an HCP structure type via PTM. It can be seen that the classification performed by the SOM reveals the same dislocation structures as the PTM algorithm, without some of the "noise" that the later induces. Although the sample presented in Fig. 13 is at a strain 50.5%, this was also the case for the sample at a strain of 55.5%.

This test took around $440s$ to complete.

V.2.1

Important: *This version was built from V.2.0 and developed in parallel with V.1.2 So all of the updates in V.1.2 are also present in this version of the software.*

Description

The normalization method was updated again. For V.2.1, the normalization of a feature column C is given by Eq. 5:

$$C_{norm} = \frac{C - C_{min}^{training}}{C_{max}^{training} - C_{min}^{training}} \tag{5}$$

Where C_{norm} is the normalized data column and $C_{min}^{training}$, $C_{max}^{training}$ are the minimum and maximum values, respectively, of the respective feature column in the training data. This normalization method generates a column with range $[0, 1]$, solving the issue caused by Eq. 1, that wasn't properly fixed by Eq. 2.

There was a considerable speed-up of the classification process, since it now implements the vectorization of the distance calculation between the data points and the output neurons.

The writing time of the output file was also greatly improved. This was achieved by moving from a Python-level loop through every line of the data to a Pandas built-in method that generates a single string with all of the data, so that it can be written more efficiently.

On the other hand, the training process of the SOM remains to be optimized.

Testing:

HEA Nano-foam Under Tension:

The same HEA nano-foam under tension from the test performed for V.1.0 was used, doing the same test as in V.2.0. The atoms on each file were classified into 6 groups. Furthermore, the SOM network was trained using parameters $\sigma = 0.2$, $\eta = 0.5$ and $f = 1$, and the features used were:

- Radial function coordination
- Centro-symmetry parameter (12 neighbors)

It's important to clarify that, although the network had 6 output neurons, the final classification of the atoms used 5 groups. Figs. 14 and 15 show the results of this test. It can be observed that the transfer learning is greatly improved with respect to V.2.0, and that the slip planes identified in previous tests on this sample are again correctly identified.

This test took around $150s$ to complete, which indicates a speed-up of around 3x with respect to V.2.0.

HEA Nano-foam Under Compression:

The same HEA nano-foam under compression from the test performed for V.1.0 was used, but with the goal of classifying the atoms into 6 groups. The SOM network was trained using parameters $\sigma = 0.5$, $\eta = 0.5$ and $f = 1$, and the features used were:

- Radial function coordination
- Centro-symmetry parameter (with 12 and 18 neighbors)

Even though 6 groups were used to classify the atoms, groups 1 and 2 contained less than 0.01% of the atoms in the sample. So it's reasonable to say that the SOM clustered the atoms into 4 effective groups. The results are shown in Fig. 16, where a comparison between the SOM clustering and the PTM algorithm is presented.

This test took around $113s$ to complete.

V.2.2

Important: *This version was built from V.2.1 and developed in parallel with V.1.3 So all of the updates in V.1.3 are also present in this version of the software.*

Description

The training process was optimized through a matrix formulation of the distance between a given data point and all the output neurons of the SOM network. This is presented in Eq. 6:

$$D^2 = X \cdot \vec{x}^T - 2(\vec{x} \cdot W)^T + diag(W^T W)^T \tag{6}$$

Where $\vec{x}$ is the data point, W is the weight matrix of the SOM, and X is a matrix defined as $X_{mj} = (\vec{x})_j, \forall m \in \{1, ..., N\}$ (N being the number of output neurons of the SOM).

It was observed that this method was slower than the previous method for $N = 2$ output neurons, so the software implements the training process of V.2.1 when $N = 2$ and the new method otherwise.

Testing:

HEA Nano-foam Under Tension:

The same test as in V.2.1 was performed, obtaining the same results. In this case, this test took around $125s$ to complete, 16% faster than for V.2.1.

HEA Nano-foam Under Compression:

The same test as in V.2.1 was performed, obtaining the same results. In this case, this test took around $94s$ to complete, 17% faster than for V.2.1.

V.2.3

Important: *This version was built from V.2.2 and developed in parallel with V.1.4 So all of the updates in V.1.4 are also present in this version of the software.*

Description

A batched learning algorithm was included as an option for the training process of the SOM. This consists of partitioning the data in batches of size B, finding the BMU for every data point in a given batch and then updating the weights of the neural network through equation 7:

$$w_{mn}^t = w_{mn}^t + \eta(t)(H_{mn}(t) - w_{mn}^t) \tag{7}$$

with $H_{mn}(t)$ defined by equation 8:

$$H_{mn}(t) = \frac{\sum_{i=1}^{B} h_m(b_i, t) x_{i,n}}{\sum_{i=1}^{B} h_m(b_i, t)} \tag{8}$$

Where b_i is the winning neuron associated with the data point $\vec{x}_i$, and $h_m(b_i, t)$ is just the neighbourhood function for the m-th neuron, with respect to b_i. The definition of $H_{mn}(t)$ was inspired by the work of Matsushita and Nishio [6].

The training of the network ends when this training process has been done for every batch of the data.

Testing

HEA Nano-foam Under Tension:

The same samples and features as in V.2.2 were used, this time analyzed with parameters $\sigma = 0.2$, $\eta = 0.5$ and $f = 1$, using $N = 6$ groups to classify the data and batched learning with a batch size of $B = 100$, obtaining different results. In this case, this test took around $109s$ to complete, 12% faster than for V.2.2.

Fig. 17 shows a comparison between one of the resulting groups of the classification and the result obtained with V.2.0 of the algorithm. It can be observed that this group, associated with the HCP atoms of the sample, has less noise than previous results.

HEA Nano-foam Under Compression:

The same sample and features as in V.2.2 were used, this time analyzed with parameters $\sigma = 0.5$, $\eta = 0.5$ and $f = 1$, using $N = 5$ groups to classify the data and batched learning with a batch size of $B = 100$. This took around $74s$ to complete, $\approx 20\%$ faster than for V.2.2.

It is important to notice that in this occasion, the results seem to be qualitatively better than those obtained with V.2.2, as presented in Fig. 18, where atoms in groups 1, 2 and 3 are shown. The clustering of the sample now distinguishes the atoms associated with defects from those associated with the surface of the material.

It is also worth noting that, contrary to what is observed in the test performed with V.2.2, there are no 'empty' groups. More specifically, the smallest group contains more than 3% of the total atoms of the sample. This could indicate that batched learning is better than serial learning at using all of the output neurons to cluster the data.

References

[1] Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. *Nature 585*, 357–362 (2020). DOI: 0.1038/s41586-020-2649-2.

[2] McKinney, W. Data structures for statistical computing in python. *Proceedings of the 9th Python in Science Conference, Volume 445* (2010).

[3] Troncoso, J. F. ClasSOMfier: A neural network for cluster analysis and detection of lattice defects. *arXiv e-prints,* (2020).

[4] Plimpton, S. Fast parallel algorithms for short-range molecular dynamics. *Journal of computational physics, 117*(1), 1-19 (1995).

[5] Amigo, N. Crystalline structure and grain boundary identification in nanocrystalline aluminum using K-means clustering. *Modelling and Simulation in Materials Science and Engineering, 28*(6), 065009, (2020).

[6] Matsushita, H., Nishio, Y. Batch-learning self-organizing map with weighted connections avoiding false-neighbor effects. *The 2010 international joint conference on neural networks (IJCNN) (pp. 1-6). IEEE*, (2010).

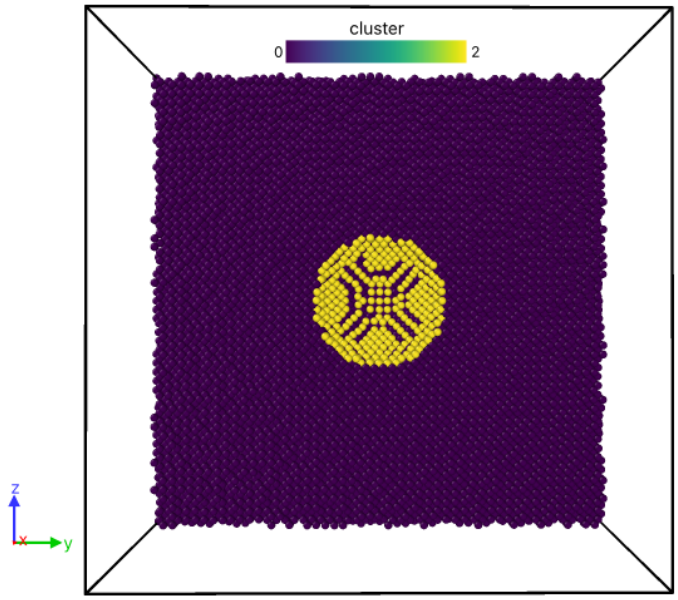

Figure 1: BCC Fe bulk with void. The atoms have been clustered into 3 groups using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$. The centro-symmetry and coordination were used as features. A slice of the sample, with groups 0 (purple) and 2 (yellow), is shown.

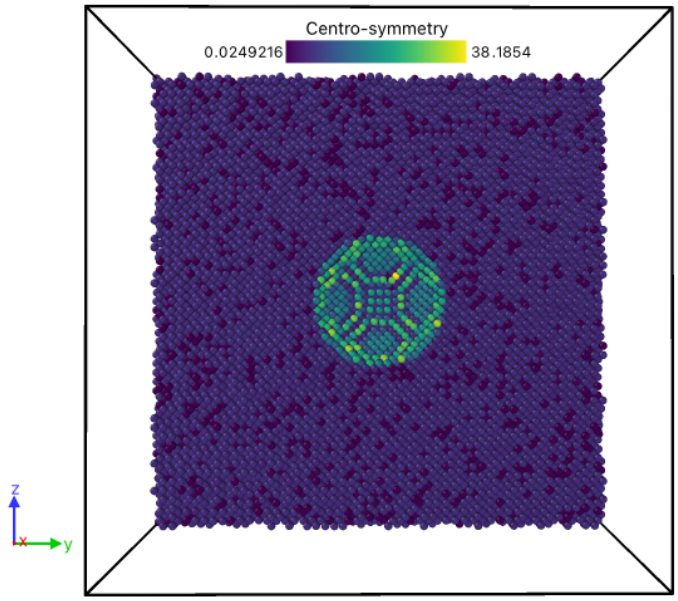

Figure 2: BCC Fe bulk with void. The atoms have been color coded using the centro-symmetry parameter (as calculated by LAMMPS). A slice of the sample is shown.

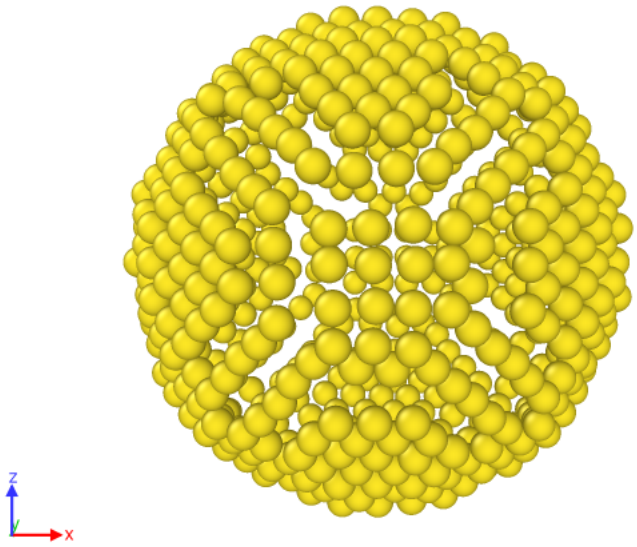

Figure 3: Void of the BCC Fe bulk sample. The atoms belonging to group 2, as classified by the SOM, are shown. The centro-symmetry and coordination were used as features.

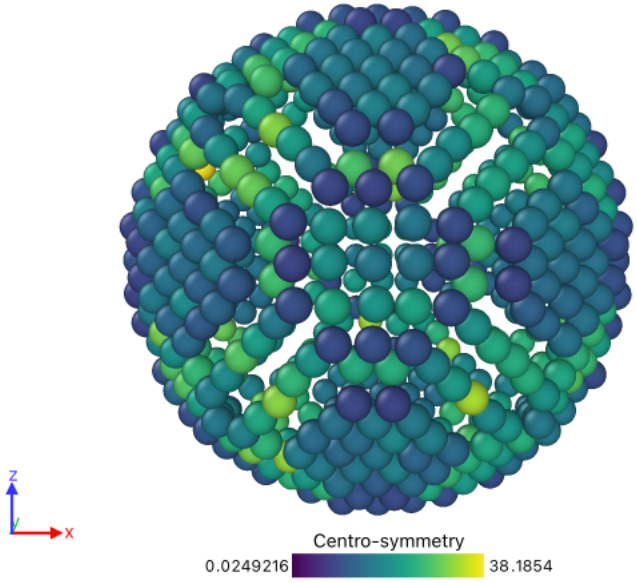

Figure 4: Void of the BCC Fe bulk sample. The atoms have been color coded using the centro-symmetry parameter (as calculated by LAMMPS).

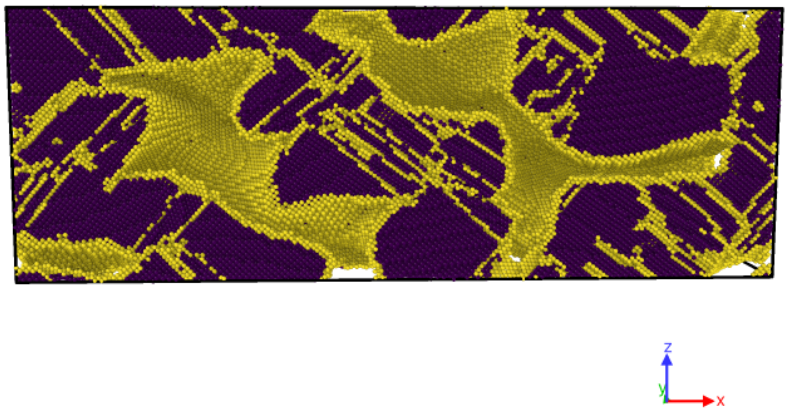

Figure 5: HEA Nano-foam under compression. The atoms have been clustered into 2 groups using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$. The centro-symmetry, coordination (via Coordination Analysis with $r_c = 6$) and atomic volume (via Voronoi Analysis) were used as features.

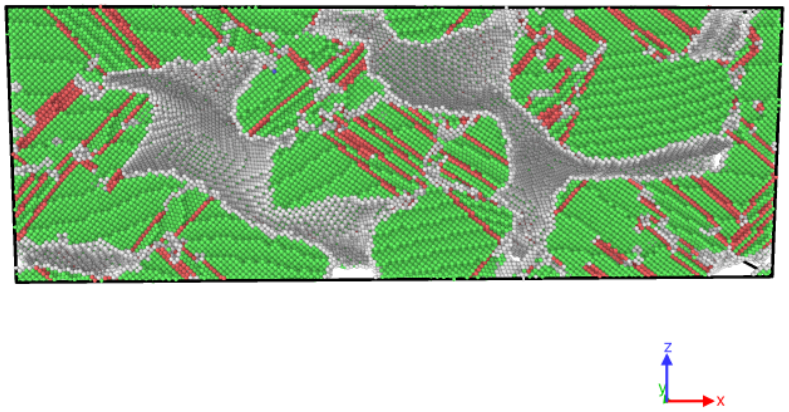

Figure 6: HEA Nano-foam under compression. The atoms have been clustered using the PTM algorithm into 4 categories: FCC (green), HCP (red), BCC (purple) and Other (white).

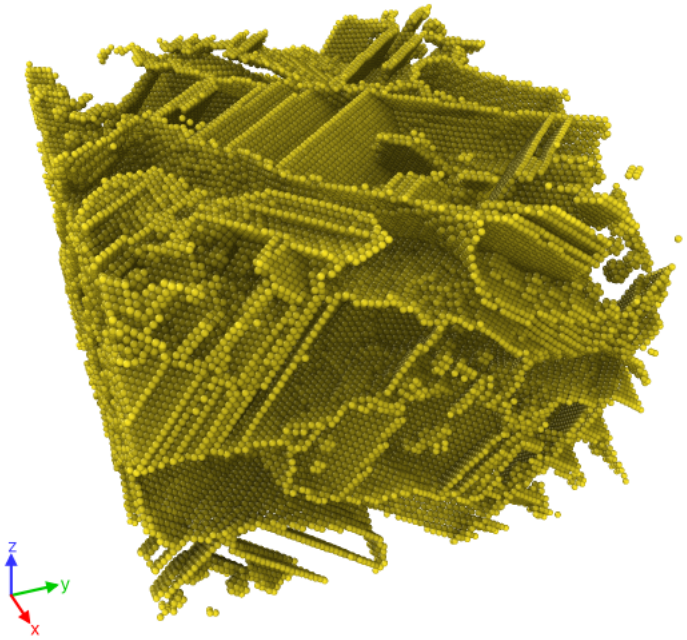

Figure 7: HEA Nano-foam under compression. Atoms classified in group 1 (yellow atoms) are shown in a slice of the sample.

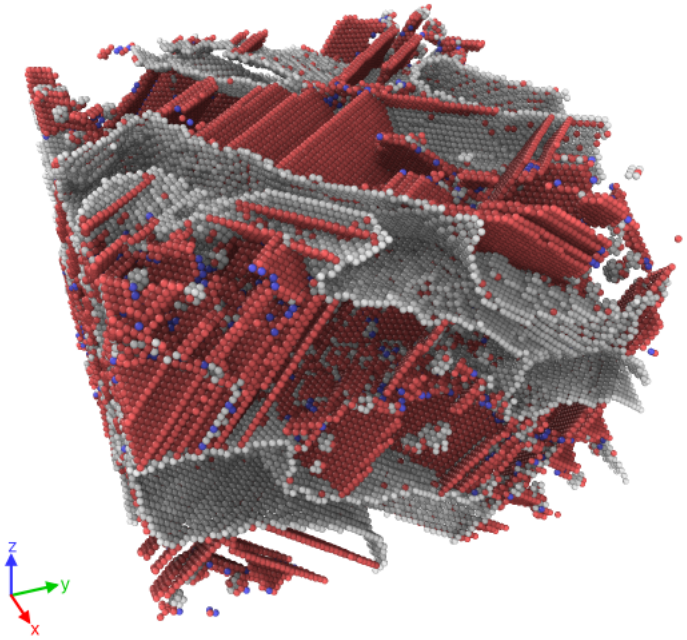

Figure 8: HEA Nano-foam under compression. Structure types HCP (red), BCC (purple) and Other (white) are shown in a slice of the sample.

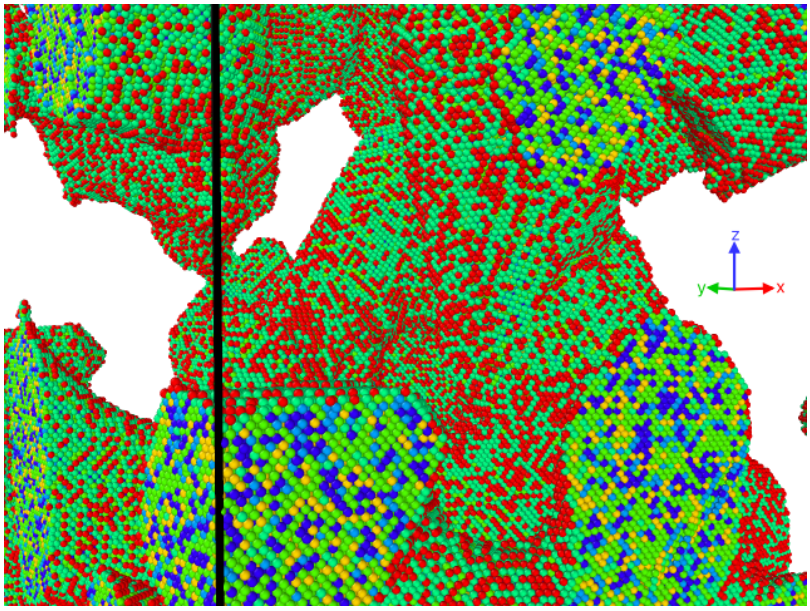

Figure 9: HEA Nano-foam under tension. The atoms have been clustered into 6 groups using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$. The potential energy, centro-symmetry parameter (with 12 and 18 neighbors), structure type, Voronoi coordination number and radial function coordination were used as features.

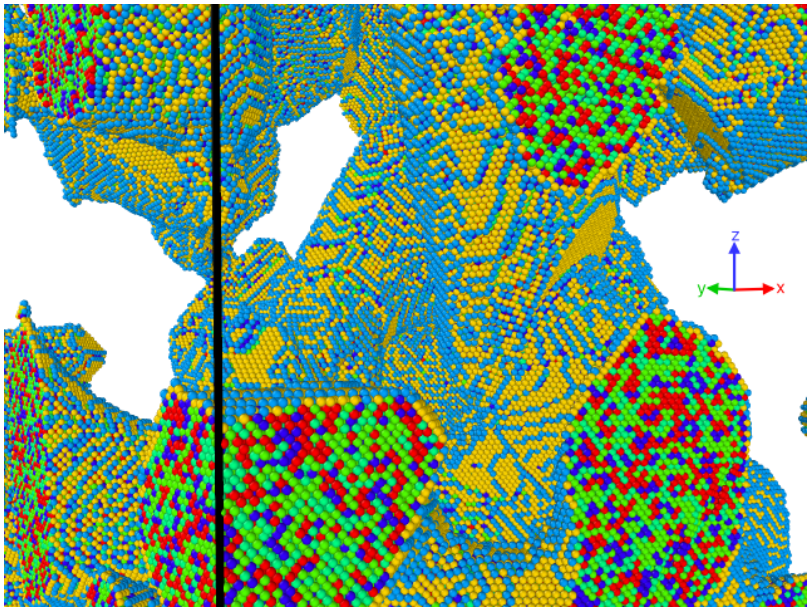

Figure 10: HEA Nano-foam under tension. The atoms have been clustered into 6 groups using N. Amigo's K-means clustering software. This analysis was performed by N. Amigo.

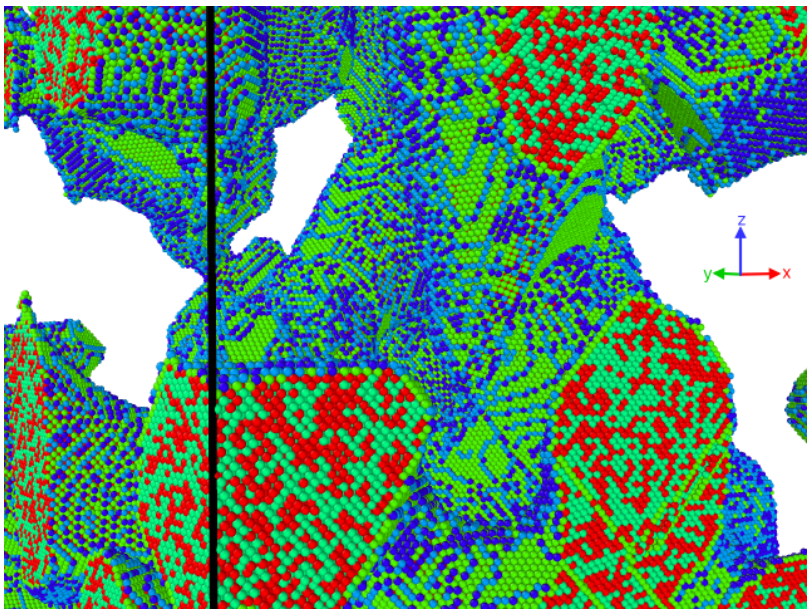

Figure 11: HEA Nano-foam under tension. The atoms have been clustered into 6 groups using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$. The atomic volume, radial function coordination and centro-symmetry parameter (with 12 and 18 neighbors) were used as features.

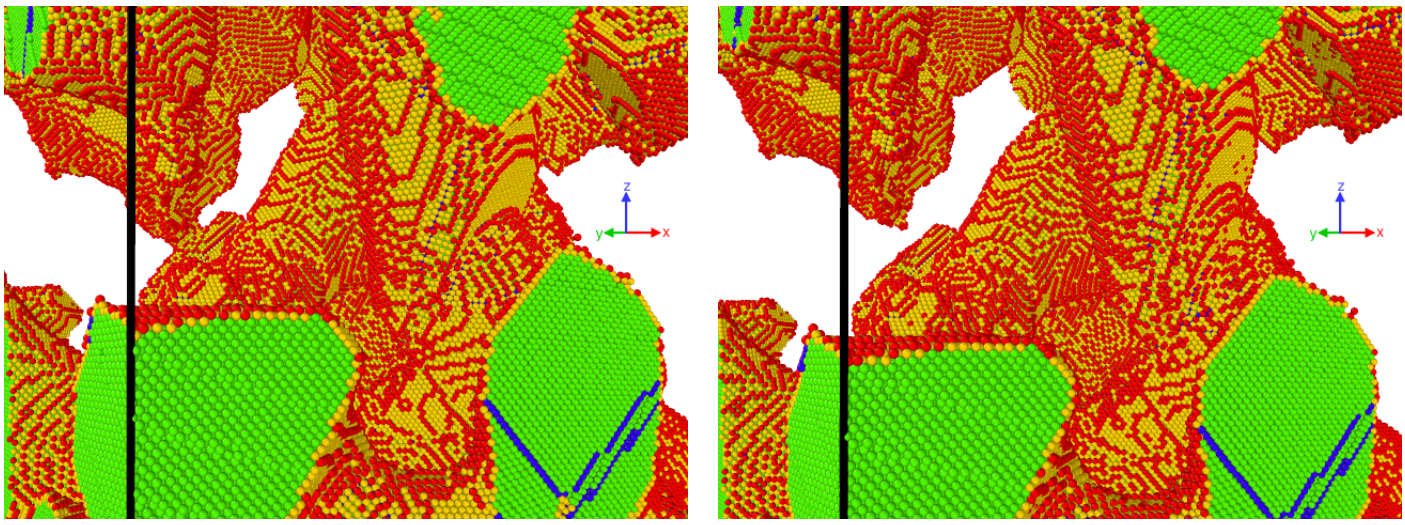

Figure 12: HEA Nano-foam under tension, at strains of 50.5% (left) and 55.5%(right). The atoms have been clustered into 6 groups using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$, and the SOM was trained with the sample at a strain of 50.5%. The radial function coordination and centro-symmetry parameter (with 12 and 18 neighbors) were used as features.

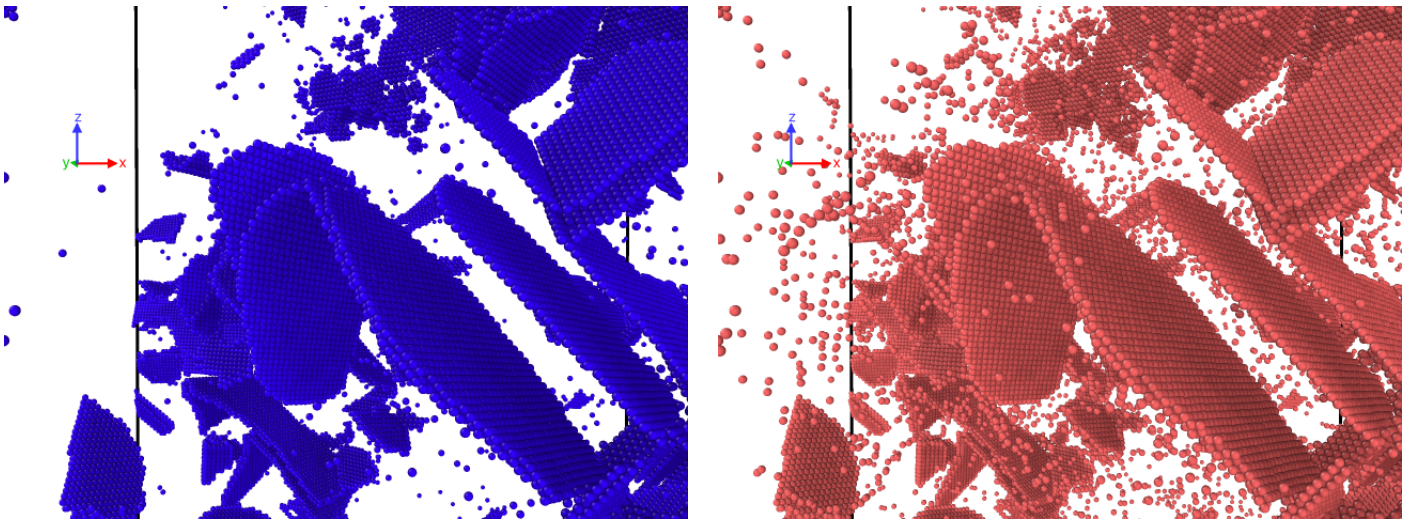

Figure 13: HEA Nano-foam under tension, at a strain of 50.5%. Left: the atoms have been clustered into 6 groups (but only one of them is shown) using parameters $\sigma = 1$, $\eta = 0.5$ and $f = 1$, and the SOM was trained with this sample. The radial function coordination and centro-symmetry parameter (with 12 and 18 neighbors) were used as features. Right: the atoms have been color coded by structure type, and those with HCP structure type are shown.

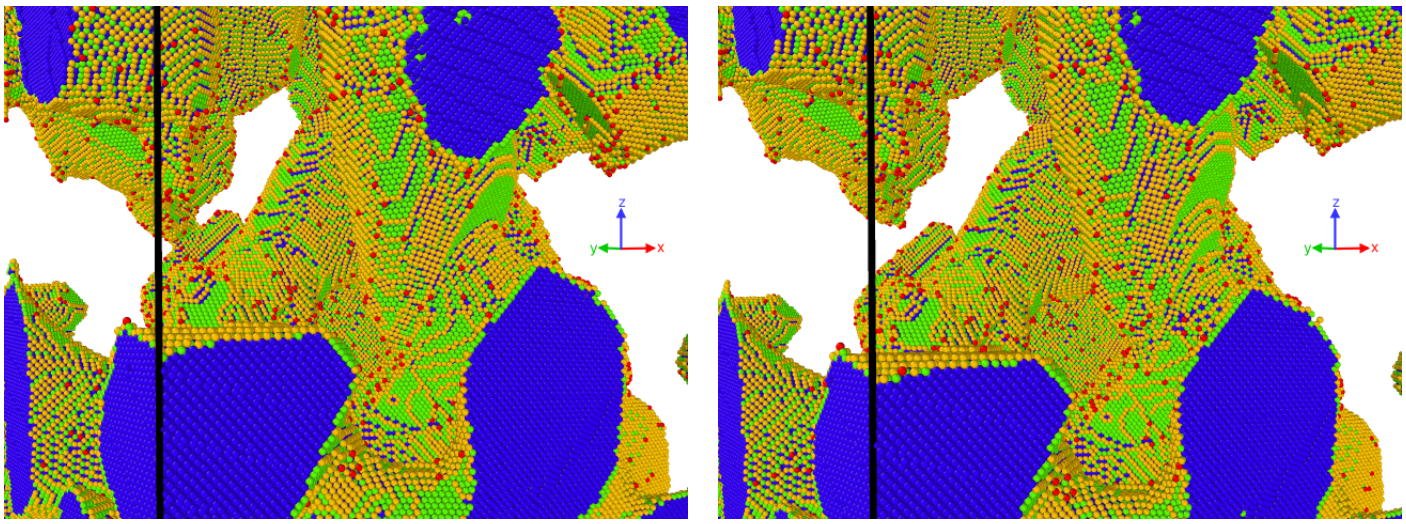

Figure 14: HEA Nano-foam under tension, at strains of 50.5% (left) and 55.5%(right). The atoms have been clustered into 6 groups using parameters $\sigma = 0.2$, $\eta = 0.5$ and $f = 1$, and the SOM was trained with the sample at a strain of 50.5%. The radial function coordination and centro-symmetry parameter (12 neighbors) were used as features. Only 5 of the 6 groups were actually used in the classification.

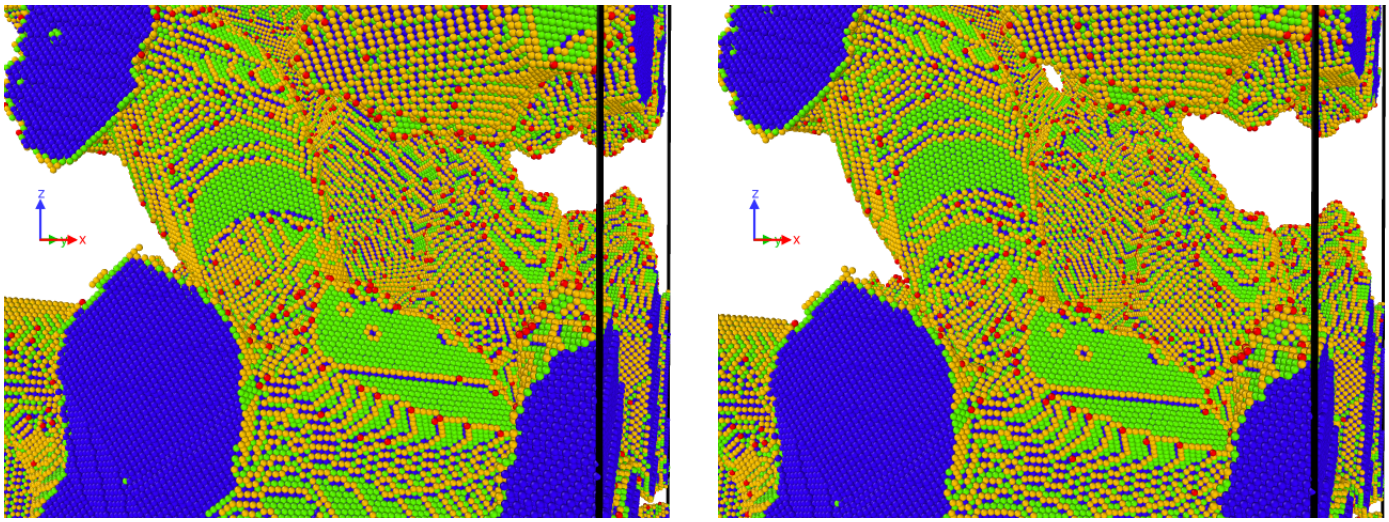

Figure 15: HEA Nano-foam under tension, at strains of 50.5% (left) and 55.5%(right). The atoms have been clustered into 6 groups using parameters $\sigma = 0.2$, $\eta = 0.5$ and $f = 1$, and the SOM was trained with the sample at a strain of 50.5%. The radial function coordination and centro-symmetry parameter (12 neighbors) were used as features. Only 5 of the 6 groups were actually used in the classification.

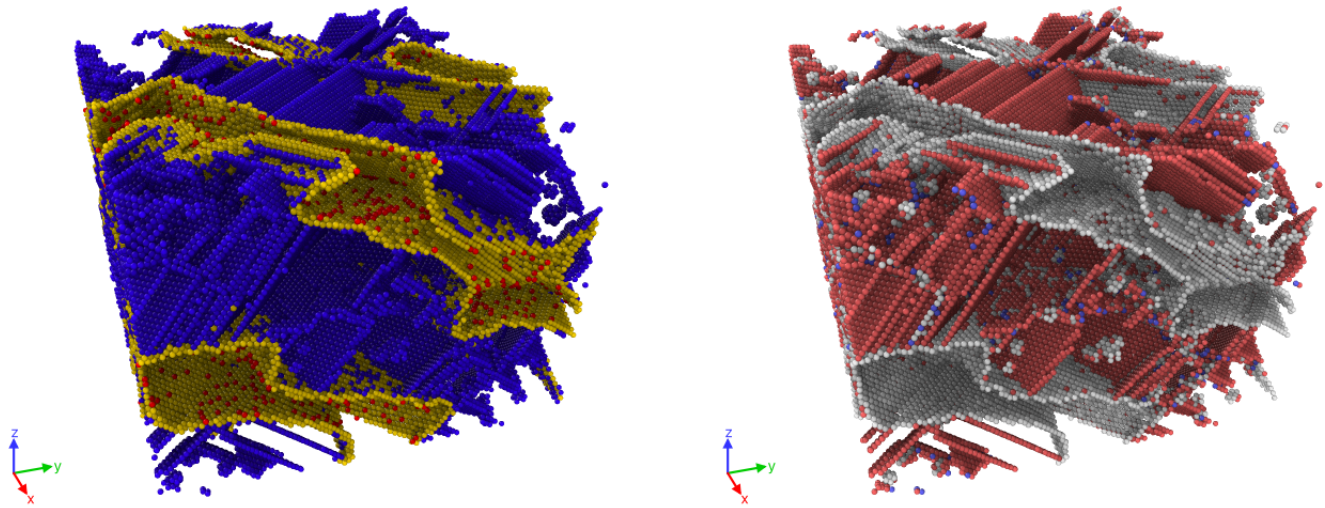

Figure 16: HEA Nano-foam under compression. The atoms have been clustered into 6 groups using parameters $\sigma = 0.5$, $\eta = 0.5$ and $f = 1$. The radial function coordination and centro-symmetry parameter (12 and 18 neighbors) were used as features. Left: The atoms classified into groups 3,4 and 5 are shown. Right: The atoms with structure types HCP (red), BCC (purple) and Other (white), are shown.

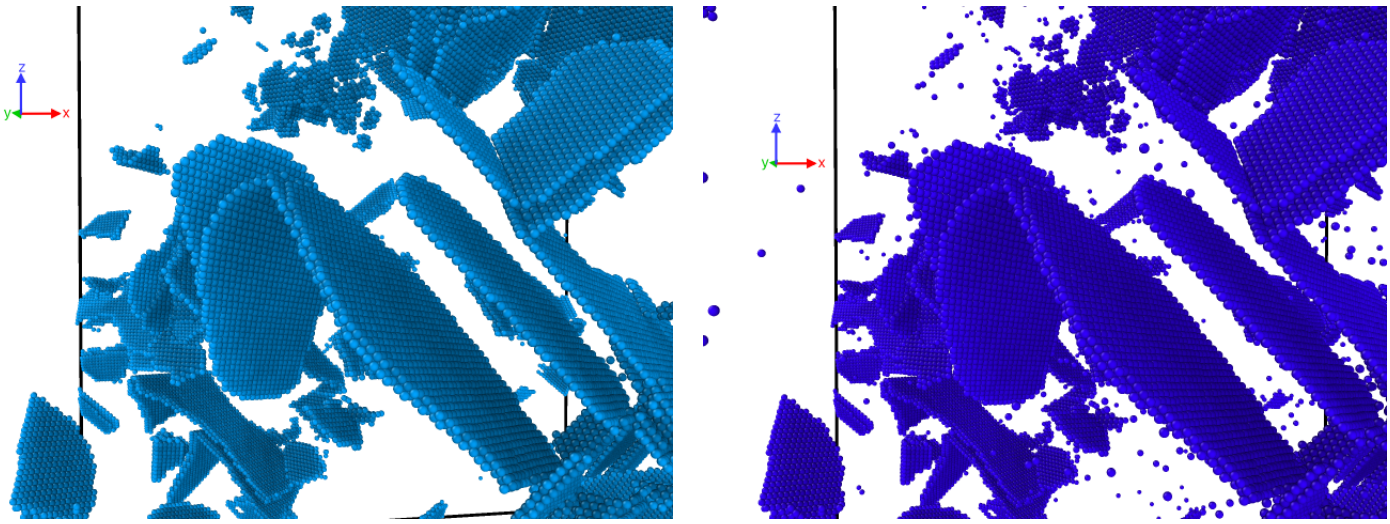

Figure 17: HEA Nano-foam under tension, at a strain of 50.5%. Left: the atoms have been clustered into 6 groups (but only one of them is shown) using parameters $\sigma = 0.2$, $\eta = 0.5$ and $f = 1$, and the SOM was trained with this sample. The radial function coordination and centro-symmetry parameter (with 12 neighbors) were used as features. Right: result previously obtained for the same sample with V.2.0.

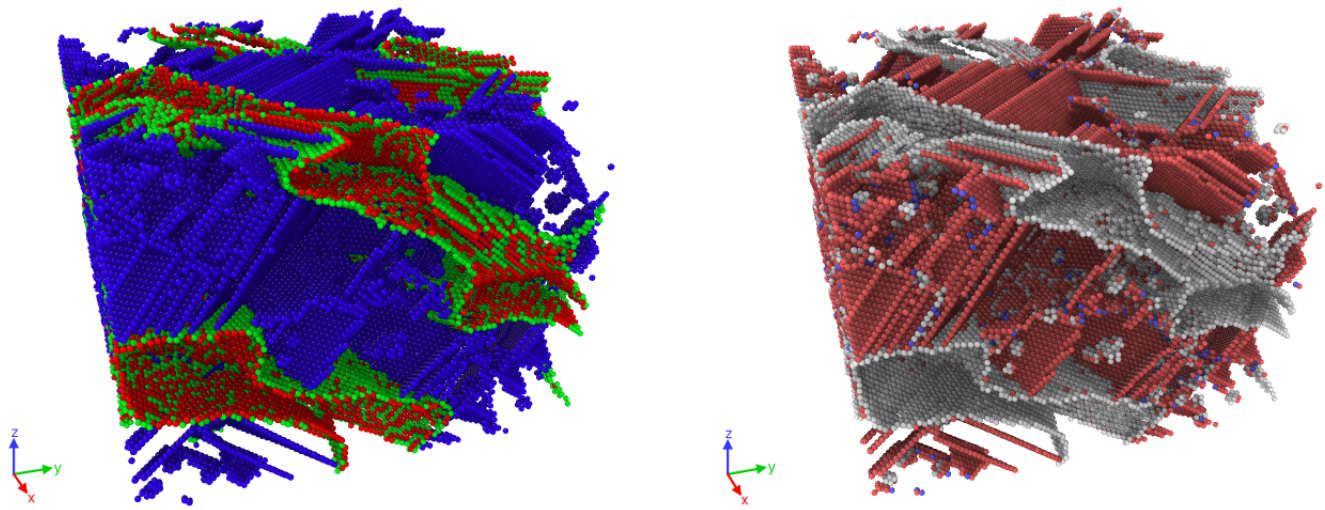

Figure 18: HEA Nano-foam under compression. The atoms have been clustered into 5 groups using parameters $\sigma = 0.5$, $\eta = 0.5$ and $f = 1$, through batched learning with a batch size of $B = 100$. The radial function coordination and centro-symmetry parameter (12 and 18 neighbors) were used as features. Left: The atoms classified into groups 1,2 and 3 are shown. Right: The atoms with structure types HCP (red), BCC (purple) and Other (white), are shown.