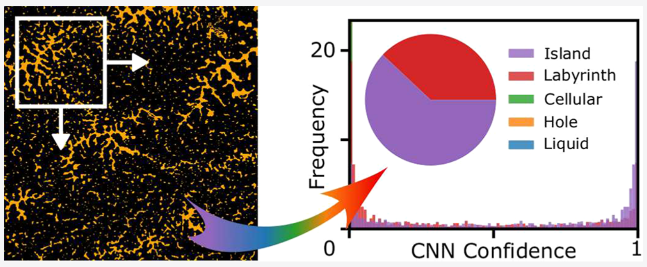
NANO LETTERS
pubs.acs.org/NanoLett
Letter

Automated Searching and Identification of Self-Organized Nanostructures

Oliver M. Gordon, Jo E. A. Hodgkinson, Steff M. Farley, Eugénie L. Hunsicker, and Philip J. Moriarty*

Cite This: *Nano Lett.* 2020, 20, 7688–7693
Read Online
ACCESS | Metrics & More | Article Recommendations

ABSTRACT: Currently, researchers spend significant time manually searching through large volumes of data produced during scanning probe imaging to identify specific patterns and motifs formed via self-assembly and self-organization. Here, we use a combination of Monte Carlo simulations, general statistics, and machine learning to automatically distinguish several spatially correlated patterns in a mixed, highly varied data set of real AFM images of self-organized nanoparticles. We do this regardless of feature-scale and without the need for manually labeled training data. Provided that the structures of interest can be simulated, the strategy and protocols we describe can be easily adapted to other self-organized systems and data sets.



KEYWORDS: *Atomic Force Microscopy, Dewetting, Nanoparticle, Monte Carlo, Machine Learning, File Search*

INTRODUCTION

There has been a recent flurry of studies applying machine learning to scanning probe microscopy (SPM), ranging from distinguishing otherwise indistinguishable data,[1] segmenting and analyzing surface features and defects,[2−6] and assessing probe quality.[7,8] While these studies will undoubtedly allow us to collect more, higher quality data in the future, we are only beginning to find ways of better using[9,10] existing data left unanalyzed on old hard drives, CDs, and even floppy discs.[11] Indeed, while one of the greatest advantages of atomic force microscopy (AFM) is its ability to produce huge numbers of scans on a variety of distinct surfaces, the need to manually select data to analyze leads to much of it being under used at best.

In this study, we demonstrate that it is possible to use simulated AFM images of self-organized nanoparticle assemblies[12,13] as automatically labeled training data for a neural network, which then correctly generalizes to real AFM scans. We employ an optimized preprocessing routine for real images of these specific structures, which is then combined with a denoising autoencoder to provide effective, automated binarisation of real images. We then combine these systems to quickly find experimental AFM images of these specific structures in a data set of 5519 scans of multiple structures and surfaces collected at a wide range of scan qualities and sizes from 500 nm to 90 μm.

SIMULATION AND CNN TRAINING

One of the biggest difficulties in employing supervised machine learning is the time-consuming task of manually labeling large training sets. Further, while scanning tunnelling microscopy (STM) and AFM scans can sometimes be simulated using density functional theory (DFT), this is only possible for relatively small system sizes and cannot practically create data at the scale required for machine learning. Instead, previous authors such as Alldritt et al.[1] and Burzawa et al.[14] used a probe-particle model involving empirical pair potentials or the Ising model, respectively, to automatically create and label training data. For the nanoparticle assemblies of interest here, we employ a highly optimized[15] implementation of the Rabani et al.[16] Monte Carlo algorithm, which accurately reproduces experimental images of 2D nanoparticle assemblies on a variety of solid surfaces.[12,13] Starting with a grid of liquid, substrate, and nanoparticles, liquid gradually evaporates based on a Metropolis[17] acceptance probability, while the nanoparticles diffuse across the surface. Over time, this produces one of a number of distinct, spatially correlated equilibrium or non-equilibrium structures. These structures have previously been variously classified as[12,18] "labyrinthine"/"fingerlike" when the nanoparticle growth is worm-like/branching, "cellular" if the nanoparticles fully enclose pockets of substrate, or conversely "islands" when the nanoparticles cluster together into isolated areas. If the solvent has not yet fully evaporated, "holes"/

Received: August 5, 2020
Revised: August 31, 2020
Published: August 31, 2020

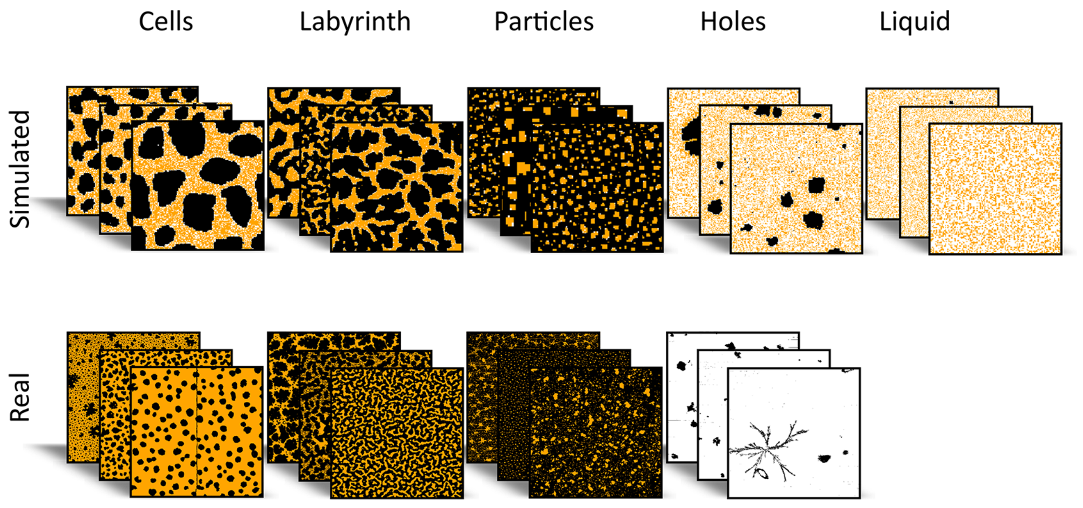


Figure 1. Monte Carlo simulations of substrate (black), liquid (white), and generic nanoparticles (orange) are used to train a convolutional neural net to classify AFM scans of nanoparticle structures. Unlike experimental images, the simulations can be programmatically classified by their Euler number (due to noise sensitivity). The trained CNN can then find the same structures in a varied data set of experimental images.

"pores" of substrate can form on a largely "liquid" surface.[12,19] Examples are shown in Figure 1.

To produce training data, we exploit the strong correlation of the initial simulation parameters with the final structure formed.[12] As such, simulations with identical input parameters almost always form images that are visually distinct yet fall within the same class of structure (cellular, labyrinthine, and so forth). Additionally, the Euler number, computed as the number of connected nanoparticle regions minus the number of connected substrate/liquid regions,[20,21] quantifies the "connected-ness" of a given nanoparticle pattern. To provide scale invariance, we then divide by the number of nanoparticles. An arbitrary, nonoverlapping range of values[12] can then be used to automatically select and label visually distinct simulated training data without prior knowledge of what parameters produce specific structures. We note that Euler classification performs poorly with real images, as it is extremely sensitive to instrumental noise, and heavily depends on the arbitrary ranges of Euler number used to define each category. Further, supervised CNNs are robust to incorrect labeling,[22] so they only require a broadly correct labeling scheme. Also, holes cannot be described by Euler numbers and so were labeled if a simulation had an abundance of liquid and isolated pockets of substrate. This information is also only fully known in simulations, and so we cannot label holes in this manner for real images.

While the automatic labeling was scale-invariant, an image-based CNN is only scale-invariant if the training images have different feature-scales. Because the visual feature size of the simulated structures depended on the simulation parameters, this was not the case by default. In particular, simulated cellular structures appeared physically larger than labyrinths, and islands were larger still. As such, the classifier would incorrectly base classifications on feature scale, rather than the features themselves. To overcome this issue, we exploited the coarsening of the nanoparticle structures from a kinetically hindered state toward their equilibrium configuration.[16,23] In regimes with fast liquid evaporation, simulations reach a metastable structure, and then grow in feature size with additional simulation steps as nanoparticles diffuse from regions of low average co-ordination to more highly coordinated sites. Larger structures therefore grow as the result of the decay of smaller features. We can therefore create different visual scales in the training data by running additional simulations for each structure type, but allowing the system to move further toward equilibrium (i.e., coarsen) by simply increasing the total simulation time. Further scale invariance was also introduced by varying simulation resolution and upscaling with nearest-neighbor interpolation to a common size of 200 × 200 pixels. Alternatively, we could instead base classification on statistics (such as the widely used[24−26] Minkowski characteristics of Euler number, perimeter, and area,[27] made scale invariant by considering number and size of particles) and a more traditional classifier instead of images and a CNN. However, we found this method to perform poorly at filtering while also being less generalizable as the statistics used must be specifically selected for the desired structures and wider data set in a manner that becomes increasingly complicated. While we therefore based our final implementation on a CNN, optimal data mining comes from thoughtfully combining machine learning and alternative statistics.

To train the classifier network, a simple CNN classifier based on the Oxford VGG[28] model was used. This model features heavily in many common machine learning applications, and a simpler implementation was used in our case as our images are discrete and lack fine detail, making the extra learning capacity unnecessary. This implementation consisted of two 2D convolutional layers with a kernel size of 32 and 3 × 3 strides and ReLu activation, followed by max pooling with 2 × 2 strides. This was then repeated with kernel sizes of 64, drop-out regularization added, and then final classification outputted with softmax activation and categorical cross entropy loss. For simplicity, we used the common Adam optimizer[29] and standard hyper-parameters[29] of $\alpha = 0.001$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and $\epsilon = 10^{-7}$. To massively augment the ~75 000 images in the training set, the periodic boundaries of the simulations allowed random circular shifts in the x- and y-axes. To simplify the learning task further, the three-level simulations were flattened to two levels, first because ideal scans have a negligible amount of liquid, and second because it is easier to binarize rather than trinarize real images. This was

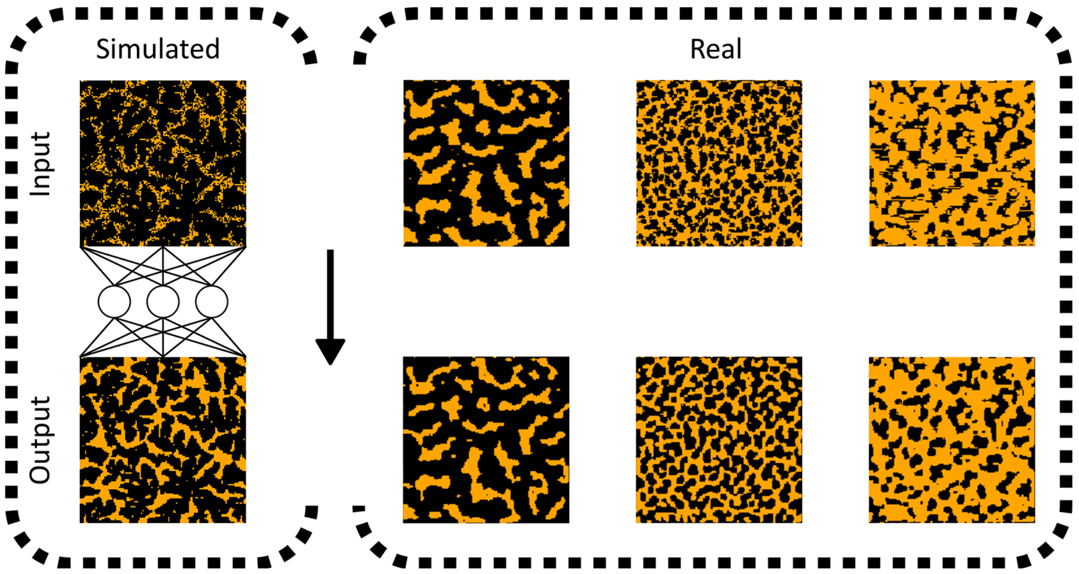


Figure 2. Results of a denoising autoencoder used to remove noise and artifacts from subsections of AFM dewetting images. Starting with simulated scans and adding speckle noise, the autoencoder removes noise by learning to recover the original simulated scan before noise was applied. This network can then be applied to real, preprocessed/binarized scans, where it is able to remove noise and artifacts from a wide variety of images.

done by replacing the least common level randomly with the other two, producing speckle noise in the process. Additional random speckle noise, level randomizing, and vertical and horizontal flips and rotations were also applied. We note that because of the relatively simple task and large variety of augmentations, only a fraction of the data was required, making the method more applicable to more intensive simulations. Because multiple model structures and hyper-parameters were not compared or tuned, performance could significantly improve by hyperparameter and structure tuning an ensemble of multiple CNNs. Because training progress was sensitive to random initialization, we did not train for a fixed number of epochs but instead until the loss on a testing set of ~2,500 images plateaued. After training, over 95% accuracy was found for a validation set of 2290 additional simulations.

■ PREPROCESSING AND FILTERING REAL DATA

After training the network on automatically labeled simulations (rather than manually labeled data), the network can then be used to find similar structures in real AFM scans even if the wanted scans are mixed with unwanted scans in a large, messy data set. The idea of using target images to find similar data is becoming of increasing interest, such as when performing reverse image searches[30] or other content-based-retrieval tasks. However, for AFM nonidentical scanning conditions means that preprocessing/binarizing data is first required before classification can take place. Because automating this process is not trivial, preprocessing is typically done manually.[31] However, unlike other automated methods[4,32] which maximize aesthetic quality for a large visual variety of images, we are not interested in aesthetic quality, and only need a broadly reasonable preprocessing method. By optimizing the routine for the target structures, we can also assume that images unable to be processed are of low quality and/or not the structures being searched for and so can be discarded.

Before classifying each image, we first applied several layers of preprocessing and filtering. After converting[33] from proprietary file format into Python (and discarding any corrupt files), we normalize and median-of-difference align the image data from the scan. Noisy scans were arbitrarily discarded if over 5% of rows had over 95% of data single-valued and/or with row mean more than 2σ from the overall image mean. A fifth order polynomial plane fit was then used for plane removal along both x and y-axes. We discarded images that were not self-similar by testing the stability of Minkowski numbers as a subsampling window was moved around the image frame. Binarization was then performed with a multiple-layered Otsu threshold,[21] which has been previously shown to be broadly optimal for AFM images.[4] We note that the shape of the structures can be significantly altered by the binarization routine,[4] resulting in the network correctly assessing the binarized image but not the actual image. However, we also note that a given image may have multiple valid binarizations and therefore multiple valid structure classifications (which itself often leads to subconscious human bias, an issue avoided by this automated approach). Regardless, classification is limited by preprocessing quality. It is also for this reason that we used multiple filters.

Further, because of preprocessing artifacts the susceptibility of CNNs to noise[34] and the inherent lack of instrumental noise in simulated training data, we found poor classification performance without denoising. To this end, we employed an autoencoder. In the same manner as a supervised CNN, an autoencoder is taught to encode an input image into a set of latent features, then decode (i.e., reverse) the encoding to reproduce the original input.[34] This semisupervised technique is often employed in anomaly detection,[35,36] as the network only learns to correctly reproduce "correct" data, so struggles to reproduce "incorrect" data. In the similar task of denoising, we input simulated data plus speckle noise (in this case 40 ± 5%) and teach it to output the original, noiseless simulations. The network therefore learns to remove noise. For simplicity, we used the above network structure (minus the final dense classification layer) for encoding, inversed this structure for decoding, and employed identical optimizer and hyper-parameters. Not only did denoising improve aesthetic quality

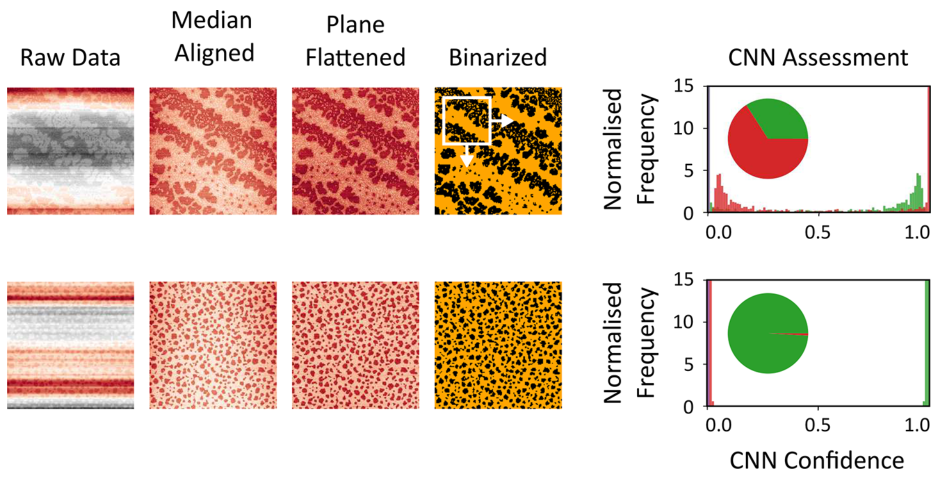


Figure 3. Automated preprocessing and CNN assessment of two AFM dewetting images of micro- and nanostructured nanoparticle assemblies on silicon substrates. Each image is subsampled and assessed with a CNN trained on simulated images of desirable structures. For the top undesirable image, the network cannot confidently pick a classification and is unsure if the structure is "cellular" (green) or "labyrinthine" (red) (among others). The bottom desirable image clearly contains cells and is correctly assessed as such by the CNN. This network, alongside other statistics, forms a system of anomaly detection and classification to find specific structures in a large, variable database.

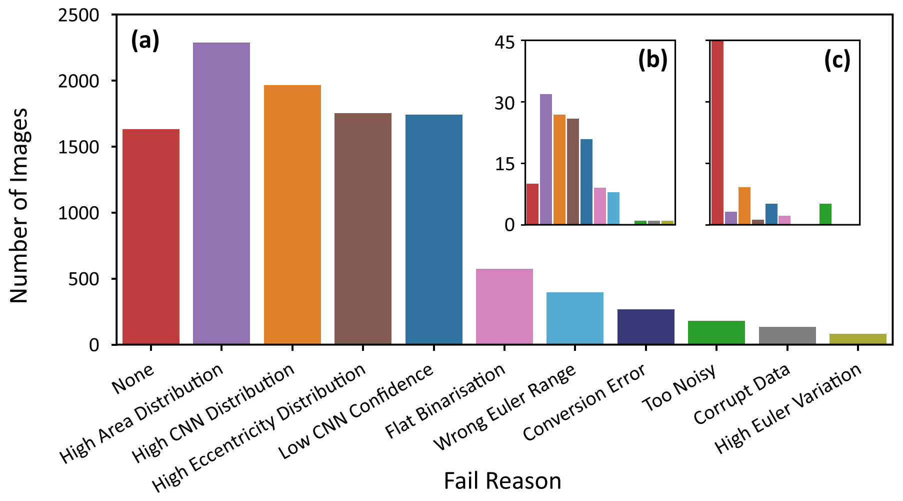


Figure 4. Filtering results on (a) a large, varied data set of 5517 scans, (b) 59 hand-selected bad images, and (c) 61 hand-selected good images. Ideally, all of the good images should pass the filter (none, red), and vice versa for the bad images. This system combines general statistical analysis with a CNN trained on Monte Carlo simulations of the target scans, and each image can fail multiple filters.

dramatically (as seen in Figure 2, in which scanning artifacts such as noise and short tip crashes were removed), but classification performance also improved as a result.

At this stage, any remaining images could then be classified by the CNN. Because the network input was smaller than the total resolution of each image, multiple assessments could be made for each single image by sliding a window around the scan frame. This is key, as it allows for images of any size to be classified. Because neural networks require fixed image resolution, new simulations and networks would otherwise be required for every different scan resolution. Additionally, this method gives a distribution of network confidences over the entire image, which can then be used as another filtering stage to remove nonhomogenous images and/or those containing different structures. Examples of this process are shown in Figure 3. To further improve filtering, receiver-operator-characteristic (ROC) curves were used to tune the threshold value used to filter/pass each image. Filtering could be further improved by simulating larger images and using the subimages as an additional input dimension when training the CNN.[8]

SEARCHING THROUGH DATABASES

To assess filtering performance, we first consider if a reasonable number of images in total are discarded by the filter. From a manual assessment of 100 random, unprocessed images (so as to avoid analysis being skewed by incorrect processing), we expected approximately 60% of images to be discarded by the filter. Despite the difficulty of the task, this was approximately the case with 70% discarded. Examples of final classifications are shown in Figure 1 with the full data set available at the end of this paper.

To determine if the correct images were discarded (i.e., not just the correct number), two data sets of 61 "good" and 59 "bad" images were then hand-selected from the final data set (without denoising). Ideally, all of the good images should not be filtered out, whereas all the bad images should be. As expected, 49/59 (83% specificity) bad images were correctly filtered out, while 45/61 (74% sensitivity) of the good images were correctly passed. Many false cases were due to poor preprocessing often because of defects or tip changes. Comparing Figure 4b,c, the filters clearly differentiated the data correctly. Additionally, many failing good images only failed a single filter, whereas many bad images failed multiple. While we chose to balance between the two, sensitivity and specificity could be adjusted by adjusting filters and their thresholds.

To assess final classification performance, we considered if a classification for an image by the entire filter/classier system was "broadly correct". This was as classification is extremely subjective and can have multiple valid outcomes. From the final output, we manually verified 20 random images from each category, finding that 15/20 holes, 14/20 cells, 13/20 labyrinths/fingers, and 10/20 islands were broadly correct. We note that as of the filtering specificity, ~20% of these misclassification were images incorrectly passing the filtering. This was also far superior to random selection from the original, unfiltered data set, in which 2/20 holes, 2/20 cells, 3/20 labyrinths, and 0/20 islands were correct. Other misclassifications were due to misleading preprocessing, were correct but nonhomogenous, or were between visually similar categories (e.g., cells and labyrinths).

Finally, while classification was clearly scale invariant (as seen in Figure 1 and information at the end of this paper), many misclassifications were made at very small feature scales. This is understandable, as at this scale the CNN convolutions would break up connected regions, effectively turning them into isolated islands and being classified as such. Furthermore, the CNNs are only as scale invariant as their training data, and our method of data simulation was unable to produce tiny feature scales while still being low-resolution enough to allow windowing. This issue is seen in areas such as painting recognition[37] and could be improved by creating an ensemble of networks with different convolution sizes trained on progressively smaller sections of each image. While outputs from the filtering/classifying system require manual verification, it can still clearly find areas of interest in data sets too large to search manually, making new analysis viable.

■ CONCLUSION

We have shown that it is possible to use simple simulations to correctly find specific structures in a data set of mixed AFM images without the need for manually labeled training data. The method is heavily reliant on good preprocessing and binarisation. While noise removal is also particularly vital, a denoising autoencoder trained on the simulated data performed extremely well at this task and may be applicable elsewhere to smooth features and remove processing artifacts. Additional improvements could be made with alternative network structures, fewer target structures, and/or a machine learning approach to binarisation,[4,38] defect segmentation, and anomaly detection.[35,36]

The method we have developed for automated identification of nanostructured patterns is an effective first stage of file search, capable of isolating files and locations of interest. It very significantly reduces the time spent searching data sets to perform additional analysis. Provided that a simple model of structure growth and a method to broadly categorize simulations is available, this CNN protocol can be easily applied to other forms of SPM, data sets, and target structures. When rapid simulation is not possible, strategies such as regression using scale-invariant statistics could also be effective with a full comparison being the subject of future work.

■ AUTHOR INFORMATION

Corresponding Author

Philip J. Moriarty − *School of Physics and Astronomy, University of Nottingham, Nottingham NG7 2RD, United Kingdom;* orcid.org/0000-0002-9926-9004;
Email: philip.moriarty@nottingham.ac.uk

Authors

Oliver M. Gordon − *School of Physics and Astronomy, University of Nottingham, Nottingham NG7 2RD, United Kingdom;* orcid.org/0000-0001-8733-7500
Jo E. A. Hodgkinson − *School of Physics and Astronomy, University of Nottingham, Nottingham NG7 2RD, United Kingdom*
Steff M. Farley − *School of Science, Loughborough University, Loughborough LE11 3TU, United Kingdom*
Eugénie L. Hunsicker − *School of Science, Loughborough University, Loughborough LE11 3TU, United Kingdom*

Complete contact information is available at:
https://pubs.acs.org/10.1021/acs.nanolett.0c03213

Notes

The authors declare no competing financial interest.
A complete data set of all preprocessed and classified images is available through the Nottingham Research Data Management Repository at http://doi.org/10.17639/nott.7067. Code is available at https://github.com/OGordon100/pyRabani/tree/CNN_Screening. AFM scans were produced with an Asylum Research MFP-3D at room temperature. Training was performed using Python 3.6.10, TensorFlow 1.15 with Keras backend, and an Nvidia Titan Xp.

■ ACKNOWLEDGMENTS

The authors thank Andrew Stannard both for the initial Monte Carlo code and the collection of historic AFM images alongside Matthew Blunt and Emmanuelle Pauliac-Vaujour. The authors also thank Michael Wilson for assistance importing the Monte Carlo simulation code from MATLAB to Python. This work was supported by the Engineering and Physical Sciences Research Council [grant number EP/N02379X/1]

■ REFERENCES

(1) Alldritt, B.; Hapala, P.; Oinonen, N.; Urtev, F.; Krejci, O.; Canova, F. F.; Kannala, J.; Schulz, F.; Liljeroth, P.; Foster, A. S. Automated structure discovery in atomic force microscopy. *Science Advances* **2020**, *6*, No. eaay6913.

(2) Rashidi, M.; Croshaw, J.; Mastel, K.; Tamura, M.; Hosseinzadeh, H.; Wolkow, R. A. Deep learning-guided surface characterization for autonomous hydrogen lithography. *Machine Learning: Science and Technology* **2020**, *1*, 025001.

(3) Borodinov, N.; Tsai, W.-Y.; Korolkov, V. V.; Balke, N.; Kalinin, S. V.; Ovchinnikova, O. S. Machine learning-based multidomain processing for texture-based image segmentation and analysis. *Appl. Phys. Lett.* **2020**, *116*, 044103.

(4) Farley, S.; Hodgkinson, J.; Gordon, O.; Turner, J.; Soltoggio, A.; Moriarty, P.; Hunsicker, E. Improving Segmentation of Scanning Probe Microscope Images using Convolutional Neural Networks. **2020**, arXiv, 2008.12371 (accessed on August 31, 2020).

(5) Ziatdinov, M.; Dyck, O.; Li, X.; Sumpter, B. G.; Jesse, S.; Vasudevan, R. K.; Kalinin, S. V. Building and exploring libraries of

atomic defects in graphene: Scanning transmission electron and scanning tunneling microscopy study. *Science Advances* **2019**, *5*, No. eaaw8989.

(6) Oxley, M. P.; Yin, J.; Borodinov, N.; Somnath, S.; Ziatdinov, M.; Lupini, A.; Jesse, S.; Vasudevan, R. K.; Kalinin, S. V. Deep learning of interface structures from simulated 4D STEM data: cation intermixing vs. roughening. *Machine Learning: Science and Technology (Just Accepted)* **2020**, DOI: 10.1088/2632-2153/aba32d.

(7) Gordon, O.; D'Hondt, P.; Knijff, L.; Freeney, S. E.; Junqueira, F.; Moriarty, P.; Swart, I. Scanning tunneling state recognition with multi-class neural network ensembles. *Rev. Sci. Instrum.* **2019**, *90*, 103704.

(8) Gordon, O. M.; Junqueira, F. L. Q.; Moriarty, P. J. Embedding human heuristics in machine-learning-enabled probe microscopy. *Machine Learning: Science and Technology* **2020**, *1*, 015001.

(9) Zhang, Y.; Mesaros, A.; Fujita, K.; Edkins, S. D.; Hamidian, M. H.; Ch'ng, K.; Eisaki, H.; Uchida, S.; Davis, J. C. S.; Khatami, E.; Kim, E.-A. Machine learning in electronic-quantum-matter imaging experiments. *Nature* **2019**, *570*, 484−490.

(10) Ziatdinov, M.; Maksov, A.; Li, L.; Sefat, A. S.; Maksymovych, P.; Kalinin, S. V. Deep data mining in a real space: separation of intertwined electronic responses in a lightly doped BaFe2As2. *Nanotechnology* **2016**, *27*, 475706.

(11) Hoffmann, P. M. *Life's Ratchet*; Basic Books: New York, 2012.

(12) Stannard, A.; Martin, C. P.; Pauliac-Vaujour, E.; Moriarty, P.; Thiele, U. Dual-scale pattern formation in nanoparticle assemblies. *J. Phys. Chem. C* **2008**, *112*, 15195−15203.

(13) Stannard, A. Dewetting-mediated pattern formation in nanoparticle assemblies. *J. Phys.: Condens. Matter* **2011**, *23*, 083001.

(14) Burzawa, L.; Liu, S.; Carlson, E. W. Classifying surface probe images in strongly correlated electronic systems via machine learning. *Physical Review Materials* **2019**, *3*, 033805.

(15) Lam, S. K.; Pitrou, A.; Seibert, S. Numba: A llvm-based python jit compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*; Austin, Texas, 15 November 2015; pp 1−6.

(16) Rabani, E.; Reichman, D. R.; Geissler, P. L.; Brus, L. E. Drying-mediated self-assembly of nanoparticles. *Nature* **2003**, *426*, 271−274.

(17) Metropolis, N.; Rosenbluth, A. W.; Rosenbluth, M. N.; Teller, A. H.; Teller, E. Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.* **1953**, *21*, 1087−1092.

(18) Moriarty, P.; Taylor, M. D. R.; Brust, M. Nanostructured Cellular Networks. *Phys. Rev. Lett.* **2002**, *89*, 248303.

(19) Ohara, P. C.; Gelbart, W. M. Interplay between Hole Instability and Nanoparticle Array Formation in Ultrathin Liquid Films. *Langmuir* **1998**, *14*, 3418−3424.

(20) Harer, J.; Zagier, D. The Euler characteristic of the moduli space of curves. *Inventiones Mathematicae* **1986**, *85*, 457−485.

(21) van der Walt, S.; Schönberger, J. L.; Nunez-Iglesias, J.; Boulogne, F.; Warner, J. D.; Yager, N.; Gouillart, E.; Yu, T. scikit-image: image processing in Python. *PeerJ* **2014**, *2*, No. e453.

(22) Frenay, B.; Verleysen, M. Classification in the Presence of Label Noise: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* **2014**, *25*, 845−869.

(23) Blunt, M. O.; Martin, C. P.; Ahola-Tuomi, M.; Pauliac-Vaujour, E.; Sharp, P.; Nativo, P.; Brust, M.; Moriarty, P. J. Coerced mechanical coarsening of nanoparticle assemblies. *Nat. Nanotechnol.* **2007**, *2*, 167−170.

(24) Mecke, K. R. *Statistical Physics and Spatial Statistics*; Springer: Berlin Heidelberg, pp 111−184.

(25) Legland, D.; Kiêu, K.; Devaux, M.-F. Computation of Minkowski measures on 2D and 3D binary images. *Image Anal. Stereol.* **2007**, *26*, 83−92.

(26) Mantz, H.; Jacobs, K.; Mecke, K. Utilizing Minkowski functionals for image analysis: a marching square algorithm. *J. Stat. Mech.: Theory Exp.* **2008**, *2008*, P12015.

(27) Martin, C. P.; Blunt, M. O.; Moriarty, P. Nanoparticle Networks on Silicon: Self-Organized or Disorganized? *Nano Lett.* **2004**, *4*, 2389−2392.

(28) Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. **2014**, arXiv 1409.1556 (accessed on June 11, 2020).

(29) Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization. **2014**, arXiv 1412.6980v9 (accessed on August 18, 2020).

(30) Krizhevsky, A.; Hinton, G. E. Using very deep autoencoders for content-based image retrieval. *ESANN* **2011**, 2.

(31) Nečas, D.; Klapetek, P. Gwyddion: an open-source software for SPM data analysis. *Open Physics* **2012**, *10*, 181−188.

(32) Delvallée, A.; Feltin, N.; Ducourtieux, S.; Trabelsi, M.; Hochepied, J. F. Direct comparison of AFM and SEM measurements on the same set of nanoparticles. *Meas. Sci. Technol.* **2015**, *26*, 085601.

(33) Somnath, S.; Smith, C. R.; Laanait, N.; Vasudevan, R. K.; Ievlev, A.; Belianinov, A.; Lupini, A. R.; Shankar, M.; Kalinin, S. V.; Jesse, S. USID and Pycroscopy − Open frameworks for storing and analyzing spectroscopic and imaging data. *Microsc. Microanal.* **2019**, *25*, 220.

(34) Goodfellow, I. J.; Shlens, J.; Szegedy, C. Explaining and Harnessing Adversarial Examples. **2014**, arXiv 1412.6572 (accessed on June 16, 2020).

(35) Sakurada, M.; Yairi, T. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis 14*; Gold Coast, Queensland, Australia, 2 December 2014.

(36) Zhou, C.; Paffenroth, R. C. Anomaly Detection with Robust Deep Autoencoders. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*; Halifax, Nova Scotia, Canada, August 15−17, 2017.

(37) van Noord, N.; Postma, E. Learning scale-variant and scale-invariant features for deep image classification. *Pattern Recognition* **2017**, *61*, 583−592.

(38) Tatum, W.; Torrejon, D.; O'Neil, P.; Onorato, J. W.; Resing, A.; Holliday, S.; Flagg, L. Q.; Ginger, D. S.; Luscombe, C. K. A Generalizable Framework for Algorithmic Interpretation of Thin Film Morphologies in Scanning Probe Images. *J. Chem. Inf. Model.* **2020**, *60*, 3387−3397.