Soft Matter

Machine Learning Analysis of Self-Assembled Colloidal Cones

Journal:	*Soft Matter*
Manuscript ID	SM-ART-10-2021-001466.R1
Article Type:	Paper
Date Submitted by the Author:	03-Dec-2021
Complete List of Authors:	Doan, David; Stanford University, Echeveste, Daniel; West Point Kulikowski, John; Stanford University Gu, Wendy; Stanford University

Machine Learning Analysis of Self-Assembled Colloidal Cones

David Doan,[a] Daniel J. Echeveste,[b] John Kulikowski [a] and X. Wendy Gu*[a]

Optical and confocal microscopy is used to image the self-assembly of microscale colloidal particles. The density and size of self-assembled structures is typically quantified by hand, but this is extremely tedious. Here, we investigate whether machine learning can be used to improve the speed and accuracy of identification. This method is applied to confocal images of dense arrays of two-photon lithographed colloidal cones. RetinaNet, a deep learning implementation that uses a convolutional neural network, is used to identify self-assembled stacks of cones. Synthetic data is generated using Blender to supplement experimental training data for the machine learning model. This synthetic data captures key characteristics of confocal images, including slicing in the z-direction and Gaussian noise. We find that the best performance is achieved with a model trained on a mixture of synthetic data and experimental data. This model achieves a mean Average Precision (mAP) of ~85%, and accurately measures the degree of assembly and distribution of self-assembled stack sizes for different cone diameters. Minor discrepancies between ML and hand labeled data is discussed in terms of the quality of synthetic data, and differences in cones of different sizes.

[a.] Department of Mechanical Engineering, Stanford University, Stanford, CA 94305, USA.

[b.] Department of Mathematical Sciences, United States Military Academy, West Point, NY 10996, USA.

* E-mail: xwgu@stanford.edu

1. Introduction

Microscopy has been a cornerstone of characterization techniques since its inception in the 17th century.[1] Microscopes are widely used to study biological objects, materials and minerals, and chemicals.[2–7] The resulting data from these microscopes are images, which need to be quantified for rigorous statistical analysis. However, this can require extensive user training or expertise.[8,9] Common outputs are the size, shape, and spacing of objects, as well as the categorization of objects by their identifying characteristics. Quantification is especially difficult for images with objects of complex shape, that are non-uniform in size, shape or spatial distribution, or closely packed or overlapping objects. For these cases, commonly used analytical methods, such as Fourier transforms to quantify periodicities, or thresholding to differentiate between objects or the object and background, may be insufficient to identify the regions of interest. Because of these limitations, achieving accurate statistics from these types of datasets is time-consuming and may be prone to large errors.

For the field of colloidal self-assembly of microscale particles, statistical analysis of self-assembled structures is needed to quantify the quality of a method or sample and understand the underlying physics. Usually, optical images are manually analyzed in order to determine the order of assembly and distribution of assembly configurations. Sacanna et al. used optical images to quantify the self-assembly distribution between different size spheres and cavities.[10] Mori et al. and Kawai et al. used optical images to quantify the order of assembly of microparticles onto templated structures.[11,12] Tigges et al. used confocal images to quantify degree of assembly and other metrics.[13] Although these works use similar metrics, such as degree of assembly and distribution of assembly configurations, quantifying these from optical images can be drastically different due to unique particle geometries (i.e., spheres vs cubes vs cones). This leads to developing specialized workflows that can only be applied to a specific particle geometry or intensive hand labeling.

Machine learning has revolutionized image processing in many areas such as biology, medicine, material science, and mechanical engineering by identifying objects within images.[14–17] While the precise output of a machine learning implementation varies based on the supplied data (e.g., segmentation or bounding boxes), the most common goal is to determine the contents of an image, as related to a defined set of classifications. Two classic examples are that of machine learning applied to text recognition and general object identification.[18,19] Utilizing large, manually labeled datasets, these applications of machine learning signaled the potential for this new computational technique to rival human image recognition. However, machine learning is possible in these cases because of the copious amounts of experimental data available. Unfortunately, many applications involving microscopy, including self-assembly, do not result in enough images for training data. In the cases where there may be enough data to train a model, labeling the data is likely still limited by time and resource constraints.

Here, we utilize synthetic data to supplement real microscopy images to enable us to use machine learning to identify objects in confocal microscopy images. We evaluate the efficacy of this approach on confocal images of densely packed self-assembled, colloidal microscale cones. These cones can form 1D nested chains. These structures pose a challenge to conventional object detection because the cones are partially obscured while assembled and the cones looks different depending on their orientation to the substrate (i.e., circular face on the substrate, curved sidewall on the substrate, nested structures in 1D chains). Because of these unique challenges, these cones are an ideal test case for determining the potential of machine learning as applied to self-assembly. Training our machine learning model on 200 synthetic images and 4 real images allowed us to achieve a mean average precision (mAP) of ~85%. The utility of our trained models was then furthered through post-processing steps to estimate the number of cones in a self-assembled structure and the total number of assembled cones in an image. We find that these estimates, while tending to be biased to underestimating, provide an accurate representation of the relative frequency of self-assembled structures of different sizes.

2. Fabrication and Assembly of Microcones

Microscale cones are fabricated on the Nanoscribe Photonic GT (Nanoscribe, GmbH) with a proprietary acrylic-based resist, IP-Dip (Nanoscribe, GmbH), and a high magnification objective (63X NA 1.40 Zeiss) according to a previously developed method (**Figure 1**).[13,20] Cones with diameters of 4.5 µm, 7 µm, and 10 µm were fabricated and self-assembled following Tigges et al.[13] The 4.5 µm particle has a nominal height of 2.5 µm and a wall thickness of ~0.25 µm. The dimensions of the 7 µm and 10 µm particles are proportional to the 4.5 µm particle. After fabrication, the particles are developed in SU-8 developer, treated with Pluronic F127 to stabilize the particles in solution, and dispersed into an aqueous solution in a glass well. A 0.7g/L concentration of 4 MDa polyethylene oxide (PEO) is then added to the solution as a depletant. The cones are allowed to assemble and are imaged after 24 hours using confocal laser scanning microscopy. A 405 nm excitation laser and 450-500 nm emission filter are used to image the particles, which are photoluminescent.

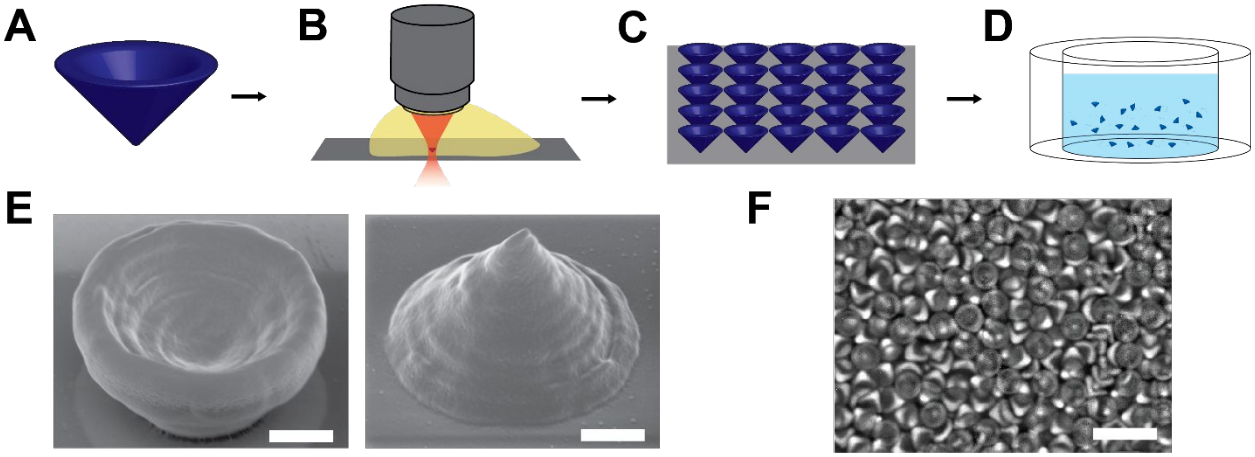

Figure 1. Process of printing and dispersing particles. A) Create a 3D model of conical shape using CAD software. B) Print particles on a substrate using 2 photon lithography. C) Resulting array of particles. D) Transfer particles into a glass well for imaging. E) SEM images of printed 4.5 μm conical particles. Scale bar is 1 μm. F) Optical image of 4.5 μm particle dispersed in a glass well after deposition. Scale bar is 10 μm.

3. Synthetic Data Generation

Synthetic data is used to generate high-fidelity labeled datasets for training machine learning models when there is insufficient experimental data for training purposes. There are many methods to generate synthetic data. For example, generative adversarial networks (GANs), traditional CGI, and domain randomization have all been used successfully.[21–23] Broadly, these methods fall into the categories of learned replication and model-centric image generation. Learned replication techniques use tools such as GANs to create synthetic images that minimize a cost function based on a set of ground-truth training data. GANs have been used to generate images for autonomous driving, facial recognition, and text recognition.[24] Model-centric image generation uses a computer simulation or image rendering software (e.g., Blender) to generate synthetic data that captures the major features of the ground-truth data specified by the user. Model-centric image generation has been previously utilized for generating images for object detection and autonomous driving.[25–28] Here, we choose model-centric image generation due to its ability to generate synthetic images based solely on prior knowledge of our target system; that is to say, it does not require the user to process data to then generate synthetic data. While there are promising projects that allow this for GAN's, labeled data is generally needed.[29] It is possible to combine multiple machine learning methods to combine their strengths, but this significantly increases the complexity and expertise needed for implementation. Furthermore, the model-centric approach is appealing since it is highly generalizable—studying a new particle geometry will not require the training of a new synthesizing network, such as with a GAN. Utilizing model-centric synthesis, we can fold the image generation process into a larger synthesis, training, and evaluation workflow.

Our machine learning workflow is seen in **Figure 2 and 3**. **Figure 2** shows the process of generating synthetic training data. This starts with creating a 3D CAD model of the particle of interest, which matches the geometry of the cones. This particle is then imported into Blender, where the particles are manually assembled into nested stacks of 1-5 particles in size (**Figure 2A**). This type of nested stack geometry is experimentally observed in our samples, and are regions of interest. In principle, the generation of stacked particles could be automated, but this would be prohibitively expensive computationally because of the non-convex shape of the particle. The next step is to generate an image with many stacks of particles with random location and orientation (**Figure 2B**). The particles are mainly oriented with the axis of the cones in the plane of the image, with a small, random, out-of-plane rotation to mimic the experimental data. This image is turned into a model of a confocal microscope image by creating slices in the imaging direction (z-slice), where the distance between the slices corresponds to the z-step size of the confocal microscope used to generate the data. These slices are convolved with a Gaussian point spread function and then added back together. Finally, Gaussian noise is added to the image. These operations are to replicate the optical processes of excitation, capture, and 2-D projection that occurs during confocal microscopy, in which point illumination is rastered in 2D or 3D to build a high-resolution image. This process is similar to a previously published method for generating confocal microscopy synthetic data.[30] Finally, the synthesized image, paired with information on the location of stacked particles within the image, is used for training data.

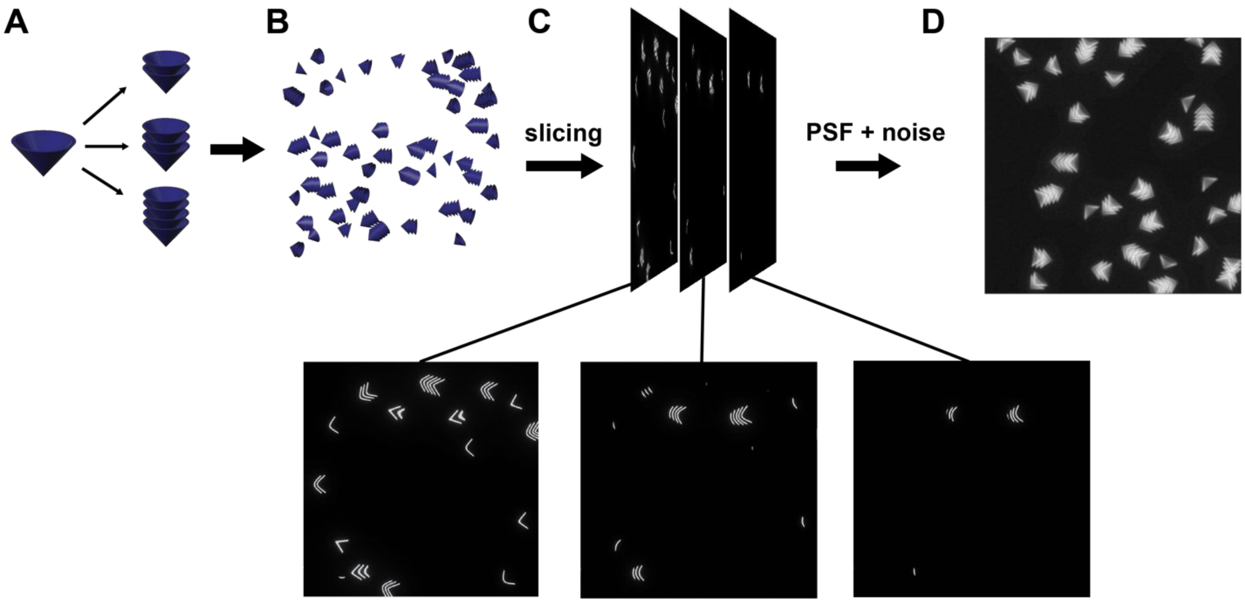

Figure 2: Generation of synthetic training data using Blender. A) A 3D CAD model of a hollow cone is loaded into Blender and assembled into stacks of nested cones. B) Single cones and stacks of cones are randomly distributed within an image. C) 2D slices of the image are rendered. D) These 2D slices are convolved with a point spread function and added back together, along with Gaussian noise to produce synthetic data that mimics confocal images.

4. Machine Learning Method

The synthetic data and a small subset of experimental data are used to train an implementation of RetinaNet for use on our target, unlabeled data (**Figure 3**). RetinaNet is a deep learning implementation that uses a feature pyramid network (FPN), a specialized convolutional neural network (CNN), to find features within an image.[31] Within RetinaNet, an additional pair of CNN's is then used to determine the bounding box and label objects based on features at various scales within the image. While RetinaNet uses this process to detect objects within an image, density estimation and point ID are two potential alternative techniques implemented in other machine learning models.[32,33] Instead of training the model to identify and label objects within an image, density estimation uses the features in an image to regress the number of particles, but not necessarily their locations. The point ID method works similarly to RetinaNet, but with object centers (instead of bounding boxes) being the target of inference. Often a method prepared for point ID can easily be converted to the regression task.[32,33] Object detection was chosen for the current study because it allows for identification of particle orientation, which greatly affects whether two adjacent particles are "related" (stacked). RetinaNet is chosen for its speed and cutting-edge performance on image localization and identification benchmarks.[31] However, in alternative implementations, models other than RetinaNet could be trained and used for inference. For example, the U-Net architecture would be reasonable for use with segmented data.[15]

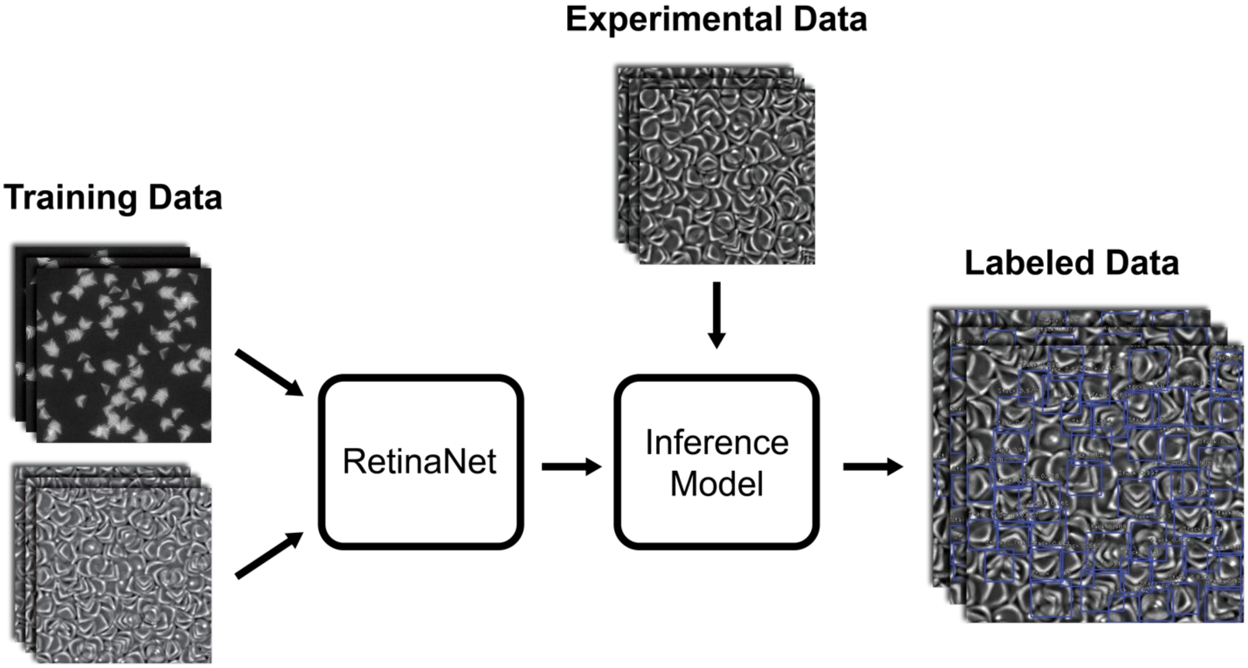

Figure 3: Diagram of model training and inference process. Synthetic and experimental images are used as inputs to train a RetinaNet model with pre-trained weights. From this training model, an inference model is generated to identify stacks of nested particles in unlabeled experimental confocal images.

5. Ablation Study on Training Inputs

The effect of modifying the number of data, type of data, and the use of pre-trained initial weights is evaluated using an ablation study. For our ablation study, a batch size of 2, an initial learning rate of 1x10^{-4}, an early-stop patience of 100 epochs, and a learning rate reduction on plateau of 1x10^{-1}, with a patience of 70 epochs is used. The only class our model was trained to identify was "stacked", as opposed to identifying different classes corresponding to the number of cones in a stack. Our standard model is trained on 200 synthetic images and 4 experimental images with the standard pre-trained ImageNet weights. Experimental images were randomly chosen for training, with at least one image of each cone size included when possible. All experimental images are preprocessed by matching the color distribution to a template image to reduce contrast variance between images. The images are then split into 612 pixel by 612 pixel sub-images. The mean average precision (mAP) for a 50% intersection over union (IoU) is estimated by validating the model over 79 experimental images. mAP is a measure of the area under the Precision-Recall curve, which is a curve plotting precision (the ratio of correct detections to total detections) against recall (the ratio of correct detections to total possible correct detections). Loss, a common measure of model performance during training, is the sum of the smooth L1 loss associated with regressing the bounding box coordinates and the focal loss, which is associated with label predictions. [31] For the standard model, the highest mAP is ~82%. This ImageNet model is then compared to a model with no pre-trained weights, and a model using weight pre-trained on Microsoft's Common Objects in Context (COCO) dataset. The mean average precision (mAP) and loss are measured per epoch for each case (see **Figure 4A**). The highest achieved mAP for no weights and COCO are ~53% and ~77%, respectively. The mAP plateaus at a training epoch of ~80 in all cases. The results of the COCO and ImageNet runs both show that our analysis can take advantage of transfer learning from more traditional datasets. The better performance of ImageNet compared to COCO is likely because ImageNet is a larger dataset that consists of more diverse categories than COCO.[19,34]

Different combinations of synthetic and experimental images are also investigated. The number of synthetic images (0, 200, 400) is varied while keeping the number of experimental images (4) the same. The number of experimental images (0, 4, 8) is then varied while keeping the number of synthetic images (200) the same. All these models are trained with ImageNet pre-trained weights. The mAP and loss are measured per epoch for each case (see **Figure 4B and C**). The highest achieved mAP is ~85% with 200 synthetic images and 8 experimental images. It is notable that the use of 200 synthetic images with 4 experimental images outperforms the model trained on 400 synthetic images and 4 experimental images. This is despite the 400 synthetic image model having a lower loss. This indicates that, above a certain threshold, the inclusion of more synthetic data leads to a degradation in performance due to overfitting. A similar effect was noted in Yao et. al. and also motivated their use of a relatively small sample of synthetic data for training.[35] The use of only experimental images (2, 4, 6, 8, 10) is evaluated in **Figure 4D**. The highest achieved mAP is ~81% for 10 experimental images, which is marginally better than 8 experimental images (mAP ~79%). This performance is comparable to the use of 200 synthetic data with 4 experimental images.

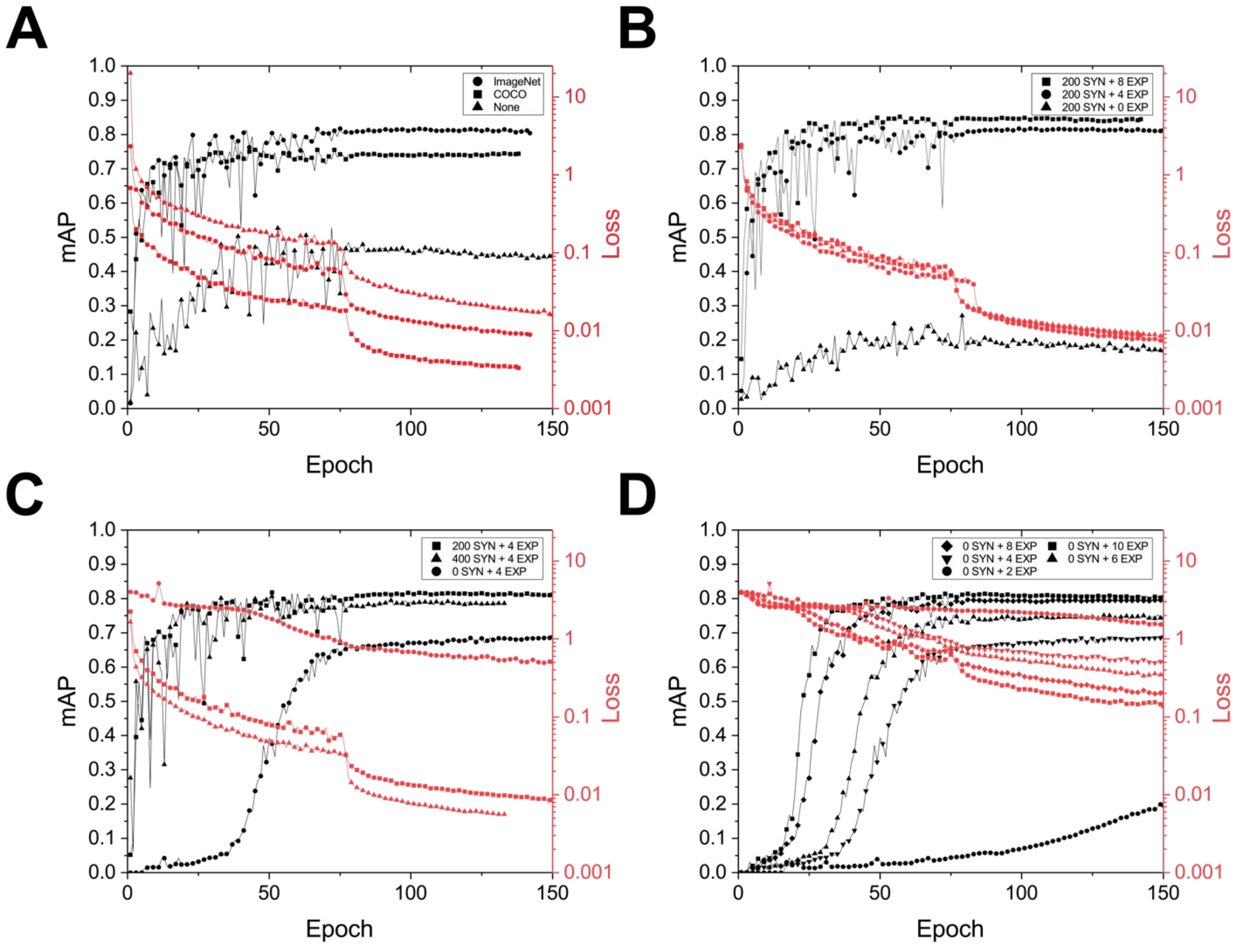

Figure 4: Mean average precision (mAP) and loss vs training epoch for **A)** 200 synthetic images and 4 experimental images with different pre-trained weights (ImageNet, COCO, and no weights). **B)** 200 synthetic images with 0, 4, and 8 experimental images with ImageNet pre-trained weights. **C)** 0, 200, and 400 synthetic images and 4 experimental images with ImageNet pre-trained weights. **D)** 0 synthetic images and 2, 4, 6, 8, and 10 experimental images with ImageNet pre-trained weights.

6. Validation and Discussion

A 100 - 200 µm square region is experimentally imaged to quantify the degree of self-assembly. This region contains approximately 900 particles for the 4.5 µm cones, 1100 particles for the 7 µm cones, and 1200 particles for the 10 µm cones. This number is estimated by calculating the density of a smaller region and extrapolating it to the rest of the image. From this data, the degree of assembly, and the stack size distribution are determined manually. Degree of assembly is defined as the total number of stacked particles divided by the total number of particles in the imaged region. The stack size distribution can be characterized using the stack number average, which is a weighted average of the conical particles in a stack divided by the total number of stacks. The degree of assembly is determined to be ~2% for the 4.5 μm cones, ~30% for the 7.5 μm cones, and ~33% for the 10 μm cones (**Figure 6A**). The stack distribution is shown in **Figure 6B, 6C, and 6D** for the 4.5 μm cones, 7 μm cones, and 10 μm cones, respectively.

The best machine learning model (i.e., RetinaNet trained with 200 synthetic images and 8 experimental images with ImageNet pre-trained weights) is used to analyze the same images. **Figure 5** shows different cone sizes labeled by the machine learning model, along with close-ups of the stack configurations. From the machine learning model, the estimated degree of assembly is ~1% for the 4.5 μm cones, ~26% for the 7 μm cones, and ~32% for the 10 μm cones (**Figure 6A**). Stacks of cones inferred by the model are categorized by aspect ratio and area to determine the stack size distribution. Labeled experimental data is used to determine the aspect ratio and average area of the bounding box for each stack size. The resulting inference bounding boxes are then binned using these metrics to determine the stack size.

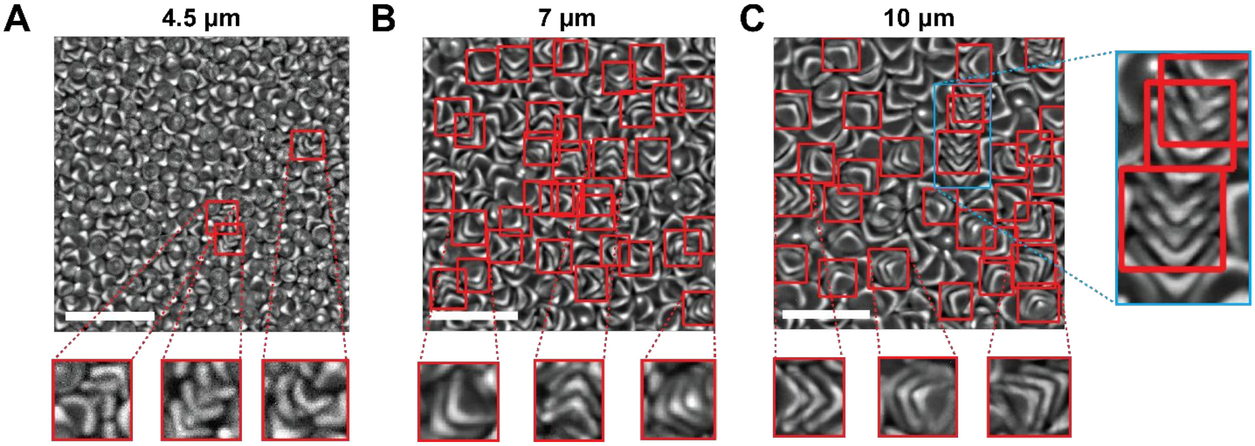

Figure 5: Machine learning model inference of stacked particles. Images of stacked particles, with closeups, identified by machine learning for the A) 4.5 μm cones, B) 7 μm cones, and C) 10 μm cones. Close up image of a longer stacked particle being identified as a combination of smaller stacks. Scale bars are 25 μm**.**

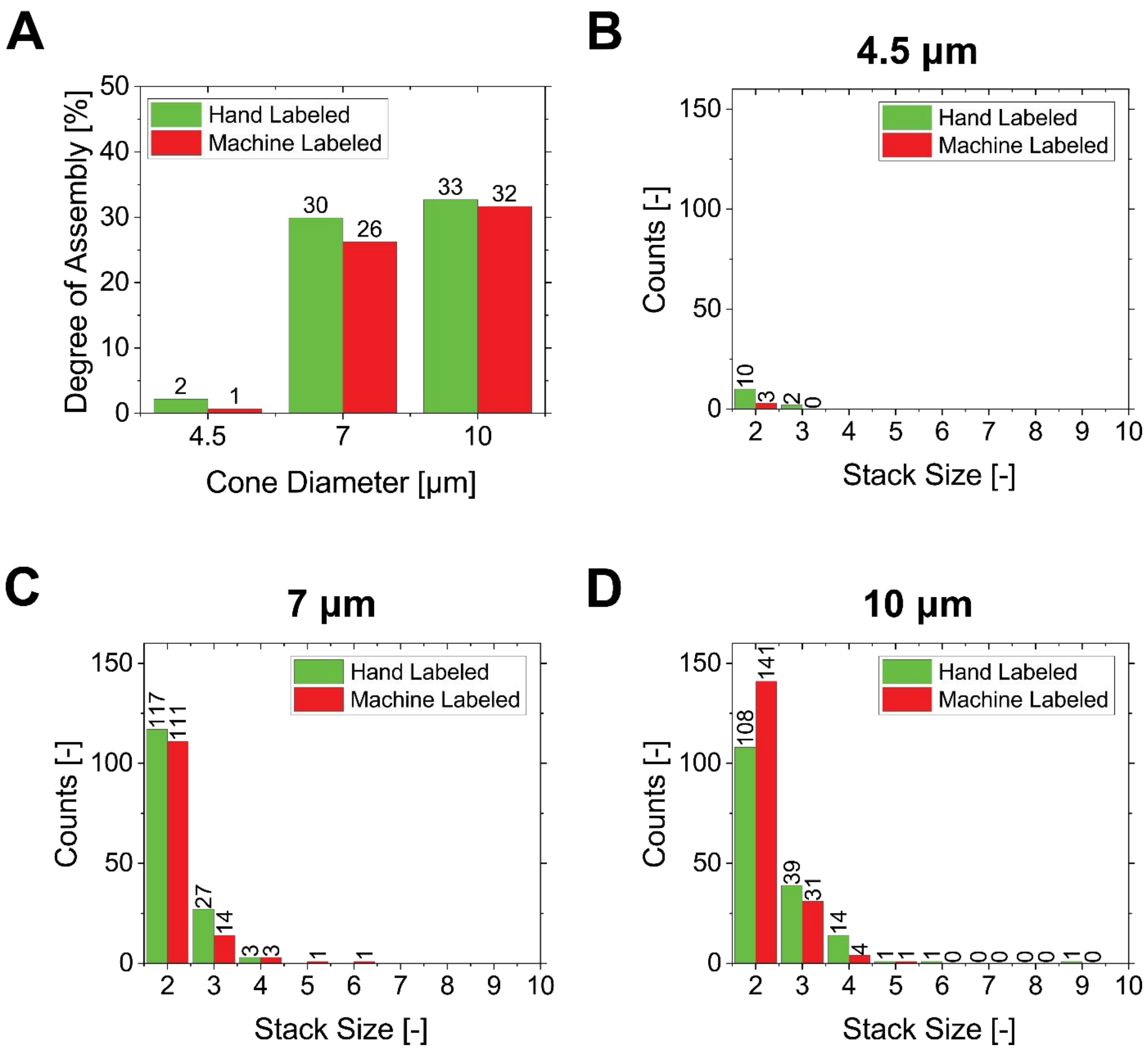

Figure 6: Histograms of hand labeled data and machine labeled data. A) Degree of assembly (%) of 4.5 µm, 7 µm, and 10 µm cones for hand labeled (green) and machine labeled (red) data. Distribution of stack size for B) 4.5 µm, C) 7 µm, and D) 10 µm cones.

The degree of assembly predicted by the model is within 2% of the manually calculated value for the 4.5 µm cone, within 4% for the 7 µm cone, and within 1% for the 10 µm cone. For all cases, both the degree of assembly and the stack average number are underestimated by the machine learning model. However, the trend of increasing degree of assembly with increasing cone diameter is captured. **Figure 6B and 6C** shows that the stack distribution seems to capture a similar number of stack instances for the 7 μm and 10 μm but severely underestimates the larger stacks. Stack instances are underestimated in the 4.5 μm case.

These errors can be partially attributed to the machine learning model inference, the synthetic data used, and the post-processing algorithm. Although RetinaNet implements FPN, which should result in a scale-invariant model, the experimental images are not scale-invariant. The machine learning model was trained on at least one image of each cone size. However, the low degree of assembly of the 4.5 μm cones led to a sparsity of labeled data for training, which makes it more difficult to accurately identify 4.5 μm cones using the machine learning model. In addition, due to the resolution of the confocal microscope, the features that the model uses to identify stacks is slightly different between the 4.5 μm cones and the larger cones. As shown in the close-up images of the identification of cones in **Figure 5**, the smaller 4.5 μm cones have a slightly different contrast profile than the 7 μm or 10 μm cones. This may account for the larger discrepancies that we see for the 4.5 μm cones. We would also like to note that since the 4.5 μm cones have a low degree of assembly, there are few objects to identify, such that missing one object leads to a large statistical difference.

The machine learning model also had difficulty identifying larger stack sizes accurately. For example, for the 10 μm particles, a stack of 9 particles was identified by hand but not by the machine learning model. **Figure 5C** shows that the machine learning model splits up the 9 stack into smaller stacks. This is because the stack has some curvature. The machine learning model cannot accurately identify this stack because curvature is not represented in the synthetic images that are generated. In addition, a stack of this size appears rarely, such that it is unlikely that a similar stack was represented in the experimental images used to train the model. Only stacks of size less than 5 were represented in our synthetic images. Additional synthetic data of large stacks with a variety of curvature would help with this issue. This motivates future work on procedurally generating the synthetic stacks due to the difficulty of generating a large amount of varied synthetic stacks by hand. Another issue with identifying large stacks is that the machine labeled data holds no information about the spatial relationship between stacks. This leads to the situation observed in **Figure 5C**, in which two stacks in close proximity, with an aligned orientation is not identified as a single stack. Addressing this shortcoming would require an alternative labeling scheme and a different machine learning model.

In addition to misidentifying larger stack sizes, the underestimation of stack size can also be attributed to the post-processing algorithm which categorizes stacks size by aspect ratio and area of the bounding box. We expect that a stack larger than 4 or 5 has a larger aspect ratio (length to width) than a smaller one. However, this does not necessarily translate to a larger bounding box aspect ratio. The larger stack can be positioned at a diagonal, making its bounding box effectively 1:1. This can be mitigated by accounting for the area of the bounding box, but the correspondence between bounding box size and stack size is not perfect, leading to the misidentification of stack size. This post-processing algorithm could be replaced with another CNN, which would classify the stack size.

7. Conclusion

In this paper we demonstrate the use of machine learning, trained on a mix of synthetic and experimental data, for the identification of self-assembled microscale cones in densely packed and noisy confocal images. We have implemented a model-based process for synthesizing training data. Through post-processing steps, we were able to obtain estimates of percent assembly within an image and the distribution of cone stack size, which was found to follow the same trends as in hand labeled data. Further improvements in object detection and accuracy could be achieved by implementing the procedural generation of synthetic images and better rendering. With improved synthetic images, the variation in the experimental data could be captured more accurately. With improved rendering, we would be able to better represent the unique elements of our experimental data in our synthetic data, allowing for more efficient learning transfer. This work shows that machine learning paired with effective synthetic data synthesis can enable the rapid and accurate quantification of microscale structures, such as self-assembled colloids.

Conflict of Interests

There are no conflicts to declare.

Acknowledgements

DD acknowledges the National Science Foundation Graduate Research Fellowship under Grant No. 1656518. JK is supported by a Stanford Graduate Fellowship. DD, JK, and XWG acknowledge funding from the Hellman Foundation, and the National Science Foundation under Grant No.

CMMI-2052251. Part of this work was performed at the Stanford Nano Shared Facilities (SNSF), which is supported by the National Science Foundation under award ECCS-1542152. Part of this work was performed at the Stanford Cell Sciences Imaging Facility.

References

1 G. McNamara, M. J. Difilippantonio and T. Ried, *Current Protocals in Human Genetics*, 2005, **46**, 1.

2 D. J. Stephens and V. J. Allan, *Science*, 2003, **300**, 82–86.

3 W. R. Zipfel, R. M. Williams and W. W. Webb, *Nature Biotechnology*, 2003, **21**, 1369–1377.

4 D. B. Hovis and A. H. Heuer, *Journal of Microscopy*, 2010, **240**, 173–180.

5 W. Hoheisel, W. Jacobsen, B. Lüttge and W. Weiner, *Macromolecular Materials and Engineering*, 2001, **286**, 663–668.

6 S. Nie, D. Chiu and R. Zare, *Science*, 1994, **266**, 1018–1021.

7 V. Vukojevic, M. Heidkamp, Y. Ming, B. Johansson, L. Terenius and R. Rigler, *PNAS*, 2008, **105**, 18176–18181.

8 F. Pesapane, M. Codari and F. Sardanelli, *European Radiology Experimental*, 2018, **2**, 35.

9 D. Shen, G. Wu and H.-I. Suk, *Annual Review of Biomedical Engineering*, 2017, **19**, 221–248.

10 S. Sacanna, W. T. M. Irvine, P. M. Chaikin and D. J. Pine, *Nature*, 2010, **464**, 575–578.

11 Y. Mori, R. Kawai, H. Suzuki, Y. Mori, R. Kawai and H. Suzuki, *Micromachines*, 2019, **10**, 428.

12 R. Kawai, Y. Mori and H. Suzuki, *Journal of Microelectromechanical Systems*, 2019, **28**, 678–684.

13 T. Tigges and A. Walther, *Angewandte Chemie*, 2016, **55**, 11261–11265.

14 T. Falk, D. Mai, R. Bensch, Ö. Çiçek, A. Abdulkadir, Y. Marrakchi, A. Böhm, J. Deubner, Z. Jäckel, K. Seiwald, A. Dovzhenko, O. Tietz, C. Dal Bosco, S. Walsh, D. Saltukoglu, T. L. Tay, M. Prinz, K. Palme, M. Simons, I. Diester, T. Brox and O. Ronneberger, *Nature Methods*, 2019, **16**, 67–70.

15 Y. Weng, T. Zhou, Y. Li and X. Qiu, *IEEE Access*, 2019, **7**, 44247–44257.

16 A. Chowdhury, E. Kautz, B. Yener and D. Lewis, *Computational Materials Science*, 2016, **123**, 176–187.

17 T. A.-Q. Tawiah, *International Journal of Advanced Robotic Systems*, 2020, **17**, 25.

18 G. Cohen, S. Afshar, J. Tapson and A. van Schaik, presented in part at *2017 International Joint Conference on Neural Networks (IJCNN)*, IEEE, May, 2017.

19 Jia Deng, Wei Dong, R. Socher, Li-Jia Li, Kai Li and Li Fei-Fei, presented in part at *2009 IEEE Conference on Computer Vision and Pattern Recognition*, IEEE, June, 2009.

20 D. Doan, J. Kulikowski and X. W. Gu, *Particle and Particle Systems Characterization*, 2021, **38**, 2100033.

21 A. Ghorbani, V. Natarajan, D. Coz and Y. Liu, *arXiv*, 2019, arXiv:1804.06516, https://arxiv.org/abs/1804.06516v3.

22 T. Baltrusaitis, E. Wood, V. Estellers, C. Hewitt, S. Dziadzio, M. Kowalski, M. Johnson, T. J. Cashman and J. Shotton, *arXiv*, 2020, arXiv: 2007.08364, https://arxiv.org/abs/2007.08364v1.

23 J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Boochoon and S. Birchfield, presented in part at *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, June, 2018.

24 S. I. Nikolenko, in *Springer Optimization and Its Applications*, Springer, 2021, vol. 174, pp. 1–354.

25 X. Peng, B. Sun, K. Ali and K. Saenko, presented in part at *2015 IEEE International Conference on Computer Vision (ICCV)*, IEEE, December, 2015.

26 P. S. Rajpura, H. Bojinov and R. S. Hegde, *arXiv,* 2017, arXiv: 1706.06782, https://arxiv.org/abs/1706.06782v2.

27 G. Ros, L. Sellart, J. Materzynska, D. Vazquez and A. M. Lopez, presented in part at *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, IEEE, June, 2016.

28 M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen and R. Vasudevan, presented in part at *2017 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, June, 2017.

29 L. Sixt, B. Wild and T. Landgraf, *Frontiers in Robotics and AI*, 2018, **5**, 9.

30 S. Dmitrieff and F. Nédélec, *SoftwareX*, 2017, **6**, 243–247.

31 T.-Y. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017, **42**, 318–327.

32 W. Xie, J. A. Noble and A. Zisserman, *Computer Methods in Biomechanics and Biomedical Engineering: Imaging and Visualization*, 2018, **6**, 283–292.

33 E. Lu, W. Xie and A. Zisserman, *arXiv*, 2018, arXiv: 1811.00472, https://arxiv.org/abs/1811.00472v1.

34 T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick and P. Dollár, *arXiv*, 2014, arXiv: 1405.0312, https://arxiv.org/abs/1405.0312v3.

35 L. Yao, Z. Ou, B. Luo, C. Xu and Q. Chen, *ACS Central Science*, 2020, **6**, 1421–1430.