

<https://doi.org/10.1038/s41524-024-01336-0>

A robust synthetic data generation framework for machine learning in high-resolution transmission electron microscopy (HRTEM)

Luis Rangel DaCosta^{1,2}✉, Katherine Sytwu², C. K. Groschner^{1,2} & M. C. Scott^{1,2}✉

Machine learning techniques are attractive options for developing highly-accurate analysis tools for nanomaterials characterization, including high-resolution transmission electron microscopy (HRTEM). However, successfully implementing such machine learning tools can be difficult due to the challenges in procuring sufficiently large, high-quality training datasets from experiments. In this work, we introduce Construction Zone, a Python package for rapid generation of complex nanoscale atomic structures which enables fast, systematic sampling of realistic nanomaterial structures and can be used as a random structure generator for large, diverse synthetic datasets. Using Construction Zone, we develop an end-to-end machine learning workflow for training neural network models to analyze experimental atomic resolution HRTEM images on the task of nanoparticle image segmentation purely with simulated databases. Further, we study the data curation process to understand how various aspects of the curated simulated data—including simulation fidelity, the distribution of atomic structures, and the distribution of imaging conditions—affect model performance across three benchmark experimental HRTEM image datasets. Using our workflow, we are able to achieve state-of-the-art segmentation performance on these experimental benchmarks and, further, we discuss robust strategies for consistently achieving high performance with machine learning in experimental settings using purely synthetic data. Construction Zone and its documentation are available at https://github.com/lerandc/construction_zone.

Machine learning (ML) methods promise to accurately and automatically analyze large datasets at high-speeds, revolutionizing our materials characterization workflows. Many state-of-the-art ML tools rely on supervised learning techniques, where models utilize large amounts of data annotated with features of interest for training. The performance of supervised ML models, like neural networks, directly depends on the contents and generating distribution of the dataset used for model training, and, importantly, such models have been shown to extrapolate poorly beyond their training datasets¹ and have limited out-of-distribution generalization behavior^{2,3}. Developing robust ML models for automated analysis of transmission electron microscopy (TEM), a versatile technique for structural and functional materials characterization at the atomic-scale, thus requires large image datasets which fully cover experimental imaging conditions and the variety of samples one has imaged. However, manually producing

sufficiently large and diverse sets of well-annotated experimental data in order to train robust, generalizable ML models can be extremely labor intensive and creates the possibility for both human and experimental biases to negatively impact model performance during deployment. With limited experimental data, it is also difficult to investigate any failures or biases of a ML workflow arising from the choice of data used.

The prohibitive cost of producing high-quality, well-annotated experimental data for supervised learning tasks makes data simulation an attractive alternative for developing effective machine learning models. Synthetic datasets produced through materials simulations offer several distinct advantages over their experimental counterparts. In particular, high-throughput simulation can create arbitrarily large datasets covering the full range of experimental conditions with ground-truth, physics-based data annotation, avoiding human bias and error in selecting and annotating

¹Department of Materials Science and Engineering, University of California Berkeley, Berkeley, CA, USA. ²National Center for Electron Microscopy, Molecular Foundry, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA, USA. ✉e-mail: luisrd@berkeley.edu; mary.scott@berkeley.edu

relevant data and at a lower cost. Synthetic data generation also enables consistently reproducible end-to-end model development workflows and the ability to precisely isolate data stream effects—both positive and negative—on trained models. The final challenge then becomes choosing suitable and sufficient datasets for developing ML models to achieve a scientific task of interest, making it important to understand how the data we use influences the accuracy and quality of the scientific inferences we make and how such data curation decisions induce practical compromises between ML model performance and model development costs.

In the study of nanomaterials, developing suitable and robust machine learning models for experimental use while only training models with simulation data requires producing accurate, experimentally-similar synthetic data for a sufficiently diverse and representative set of atomic structures, as dictated by experimental needs. Recent applied ML advancements for TEM have successfully utilized simulated training data to train neural network models to analyze crystalline scanning TEM (STEM) and STEM diffraction data^{4,5}, segment and analyze high-resolution TEM (HRTEM) of 2D material structures and nanoparticles⁶, and denoise HRTEM micrographs⁷. These prior achievements, enabled by modern TEM simulation methods which can perform large-scale, high-throughput, experimentally realistic simulations across a variety of modalities^{8–12}, have been limited in scope to just a few types of atomic nanostructures^{5–7} or to only periodic crystals⁴. The narrow set of atomic structures used for training data fundamentally narrows the application scope of their models and workflows. To lift these scope limitations and enable a wider range of experimental use cases, we need flexible, atomic-structure generation tools for simulation that better mimic the variety of complex structures seen in experimental data. However, modern software tools for computational materials science tasks^{13–15} are not primarily designed for complex structure generation and do not facilitate the precise description and generation of arbitrary distributions of complex, nanoscale atomic objects, and thus inhibit sampling sufficiently numerous and diverse atomic structures in training datasets.

In this work, we develop an end-to-end workflow for training ML models to perform atomic-resolution image segmentation of experimental HRTEM data of nanomaterials using only large, high-quality synthetic datasets comprising complex, defected atomic nanostructures sampled via a robust structure generation tool. To achieve this, we develop Construction Zone (CZ) (Fig. 1), an open-source software package which enables algorithmic and high-throughput sampling of arbitrary atomic structures, which is then combined with HRTEM simulation to generate metadata rich

databases with physics-based supervision labels. We demonstrate that CZ can be used as a core component in a flexible, all-purpose simulation framework for producing high-quality synthetic materials structures from complex distributions while still offering complete control of the structure generation process

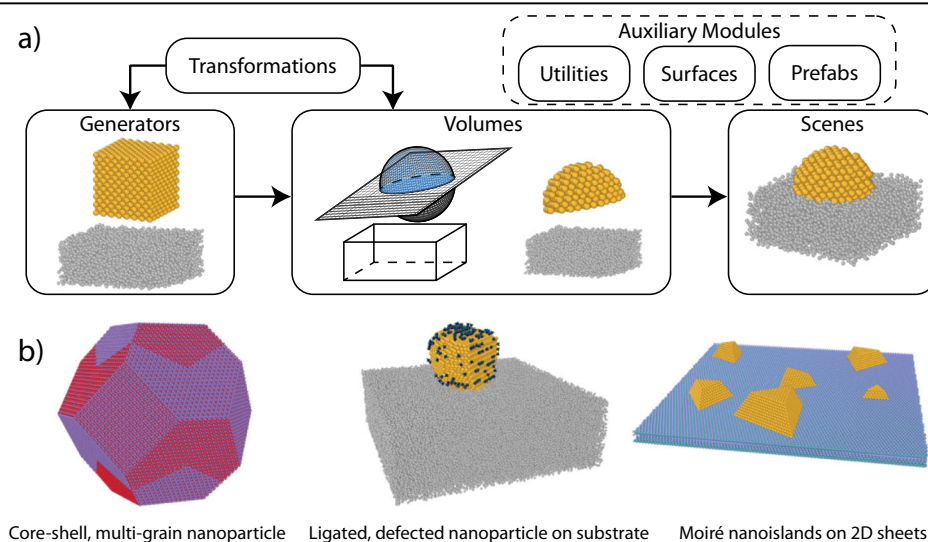
After, we narrow our focus to image segmentation of nanoparticle systems on amorphous substrates—for which we have well-described, comparable experimental benchmark data—in order to carefully study the data-dependent behavior of ML models used in HRTEM analysis. Image segmentation is a common pre-requisite for a variety of nanomaterial characterization tasks in HRTEM, used to both identify and quantitatively describe the size and shape of regions of interest in nanomaterials such as crystalline regions, crystal structures, atomic interfaces, and atomic defects. Compared to natural images, or even TEM images with directly interpretable contrast such as those taken with HAADF-STEM, HRTEM micrographs taken at ultra-high magnification have complex contrast features which cause classical image segmentation approaches to fail¹⁶. Utilizing our data curation framework, we study statistically precise relationships between aspects of the training data—including simulation fidelity, structural composition, and diversity of imaging conditions—and image segmentation performance, aggregating the performance results of several hundreds of neural networks. We benchmark performance of our neural-network based image segmentation models on a series of experimental HRTEM micrographs of clusters of Au and CdSe nanoparticles imaged at ultra-high magnification and evaluate data curation strategies and data-efficient training methods for achieving state-of-the-art neural network performance.

Results

Atomic structure and HRTEM image database generation

Models developed with ML methods for TEM analysis need to be adapted to the specific atomic structures one plans or expects to analyze in experiments; therefore, the training dataset should include examples of a wide number of likely atomic structures, so that any experimentally observed atomic structures lie close to the distribution of structures used for training. When curating a dataset to train robust HRTEM analysis models, one needs to be able to effectively and simultaneously capture both broad, high level aspects of atomic structures alongside the fine details that can be used to fully describe an individual structure. For example, an atomic structure may belong to a broader family of related structures, such as core-shell nanoparticles with similar sizes and chemistries. The same structure can also be

Fig. 1 | Diagram of the modular structure of Construction Zone. **a** Atoms are supplied by Generator objects; subtractively removed into convex objects by Volume objects; and combined together into Scenes, in which multiple objects interact. A Transformations module provides both standard symmetry operations and more complex modifications to Generators and Volumes. The rest of the package includes miscellaneous utilities for interfacing with other software, premade structural archetypes, and tools for surface analysis and modification. **b** Example structures generated in Construction Zone, including a multi-grain core-shell oxide nanoparticle with strain-mediated grain alignment⁴² (left), a heavily-faceted gold nanoparticle on a carbon substrate decorated with molecule ligands (center), and a series of gold nanoislands on a bilayer of MoS₂ forming Moiré heterostructures⁴³ (right). The Construction Zone package is available on Github at https://github.com/lerandc/construction_zone, with documentation available at <https://construction-zone.readthedocs.io/>.



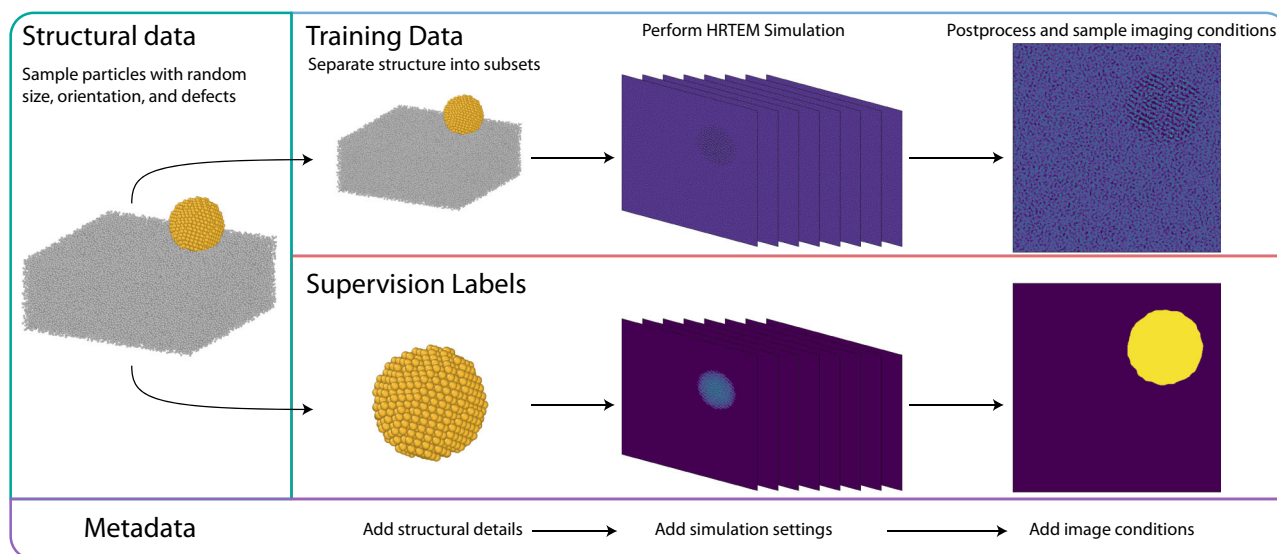


Fig. 2 | Data processing pipeline from generated structures to training-ready data. For each structure, we separate the structure into subsets for the training data and supervision label, simulate the HRTEM image formation for each structure

with Prismatic⁸, and use post-processing to sample imaging conditions, noise, and generate the segmentation mask. Metadata are accrued at each step, and the data are stored into a series of staged databases.

described with unique and specific details, such as the placement of a defect plane or the orientation of its lattice with respect to the electron beam. To fully capture and utilize these complementary structural details, we have developed Construction Zone (CZ), an open-source Python package for building arbitrary atomic scenes at the nanoscale. CZ is designed to be robust to general use-cases, such that any complex nanostructure can be made, whilst also facilitating a flexible, programmatic workflow, such that large distributions of similar objects can be generated quickly and easily, and structure generating code is both easy to interpret and easy to reuse and repurpose.

CZ relies on a simple module structure (c.f. Fig. 1a) that combines atomic placement (Generators), nano-object creation (Volumes), and nano-object interaction (Scenes). Structures can be further manipulated or generated with the Transformation class, which contains methods like standard symmetry operations, or by using convenience routines and analytical tools from the auxiliary modules, including functionality for atomic surface analysis and modification. The package derives some of its core functionality from other open-source materials science software packages, namely, PyMatgen¹³, the Atomic Simulation Environment¹⁴, and WulffPack¹⁵, and interfaces seamlessly into common simulation workflows. By allowing users to specify atomic features like defects and zone-axis orientations with a high-level, materials-focused interface, CZ enables easier generation of both specific and random nanoscale atomic structures. We showcase some example complex atomic structures ranging from nanoparticles to 2D heterostructures, generated entirely with CZ, in Fig. 1b. In our study, CZ allows us to sample and quickly generate a large number of random, similar nanoparticle structures with complex defects and varying zone-axis orientations, mimicking the collection of nanoparticles that might be imaged in a typical HRTEM experiment, whilst also tracking such metadata about each structure, enabling us to draw fully specified training data distributions for machine learning model development.

Here, we use CZ as a random structure generator alongside high-throughput TEM simulation to create large, synthetic datasets which we use to train neural networks via supervised learning techniques. These trained neural networks are then evaluated on the benchmark task of nanoparticle segmentation on experimental HRTEM micrographs of Au and CdSe nanoparticles on amorphous carbon substrates¹⁶. For each image, the goal is to classify each pixel as either part of a nanoparticle or substrate. Due to the subtle, complex interplay of contrast effects under HRTEM imaging at atomic resolution, classical image analysis techniques like Fourier filtering

fail to segment nanoparticles accurately, and are outperformed by neural networks trained on manually-labeled experimental micrographs¹⁶. Unsupervised techniques, such as using k-means clustering for pixel classification based on intensity, have only been successfully deployed for TEM images of nanoparticles taken at lower magnification¹⁷, where amplitude contrast dominates the signal, atomic lattice texture is not present, and data tend to be less noisy, thus further motivating high-accuracy supervised learning methods as a more successful route for image segmentation at atomic resolution.

To generate our synthetic dataset for training neural networks, we built a data generation pipeline, diagrammed in Fig. 2 and detailed more thoroughly in Supplementary Fig. 1, that begins with randomly generating several thousand spherical Au nanoparticles, placed atop unique amorphous C substrates, with random radii, orientations, locations, and with possible twin defects or stacking faults, to account for structural diversity in the target micrographs. We utilize Prismatic^{8–10} to simulate HRTEM images using the multislice algorithm¹⁸ and calculate their corresponding, ground-truth supervision labels, i.e., the sets of pixels in the images where nanoparticles are located. For each structure, we simulate HRTEM output waves at 300kV with a final resolution of 0.02 nm per pixel. From each simulation, multiple images of each nanoparticle structure are sampled under varying image conditions and noise. In order to facilitate targeted data curation when training neural networks, we extensively track and aggregate metadata at each phase, so that specific distributions of simulated data can be easily drawn from the full database.

Image segmentation of nanoparticles with supervised neural networks

Utilizing our data generation pipeline, we examine how aspects of the training set of simulated HRTEM images, as induced via data curation, affect neural network segmentation performance on experimental HRTEM images. By evaluating both general trends and more granular effects of dataset characteristics on neural network performance, we identify data curation strategies for training high-performance ML models for experimental HRTEM image segmentation using only simulated data. To isolate the effects on model performance due to characteristics of the training dataset, we fix our neural network architecture and optimization hyperparameters and train multiple UNet networks¹⁹ for each data condition. For a given data condition, training data are drawn I.I.D. from the simulation database, such that each network has both unique random initializations of

Table 1 | Best performance from neural networks on segmentation of nanoparticles in HRTEM images after training on various sets of simulated data, as measured on three experimental datasets

Training dataset		F1-score on Exp. Data			$N_{\text{Epochs to F1}}$	
Dataset	N_{Images}	5 nm Au	2.2 nm Au	2 nm CdSe	V_{90}	V_{95}
Baseline	512	0.710	0.740	0.681	19	21
Thermal	512	0.727	0.767	0.621	20	23
All simulation effects	512	0.822	0.814	0.647	21	24
Smaller NPs	1024	0.833	0.809	0.673	11	13
Varying Substrate	1024	0.885	0.842	0.620	11	12
Optimized Au	8000	0.915	0.808	0.648	3	3
Optimized mixed Au/CdSe	8000	0.884	0.863	0.731	1	2
Optimized CdSe	8000	0.799	0.852	0.752	2	24

In the rightmost columns, we record the median number of epochs taken for networks to reach validation F1 scores of 0.90 and 0.95 on simulated data during training.

learnable weights and independent data streams. Details regarding the composition and relevant sampling routines of each training dataset are described in Supplementary Section 1, alongside a small selection of example images drawn from our training datasets (c.f. Supplementary Fig. 3).

In order to understand our model performance in the context of the intrinsic variability of HRTEM data, we benchmark our neural network performance against three previously published atomic-resolution experimental datasets^{16,20}, taken on two different aberration-corrected TEMs by different operators at ultra-high magnification (about 0.02 nm/pixel), because contrast mechanisms in HRTEM are highly dependent on sample thickness, chemical composition, structure, and experimental conditions. The first dataset comprises images of large Au nanoparticles (5 nm) and agglomerates²¹; the second small Au nanoparticles (2.2 nm)²²; and the final small CdSe nanoparticles (2 nm)²¹. Simultaneous benchmarking on several different experimental datasets provides an opportunity to analyze how robust our trained models are to distributional shift and to determine the generality of training data effects. Previous best results on these datasets, when analyzed with neural networks trained with experimental data, are F1 scores of 0.89, 0.75, and 0.59, for the large Au¹⁶, small Au²⁰, and CdSe¹⁶ datasets, respectively.

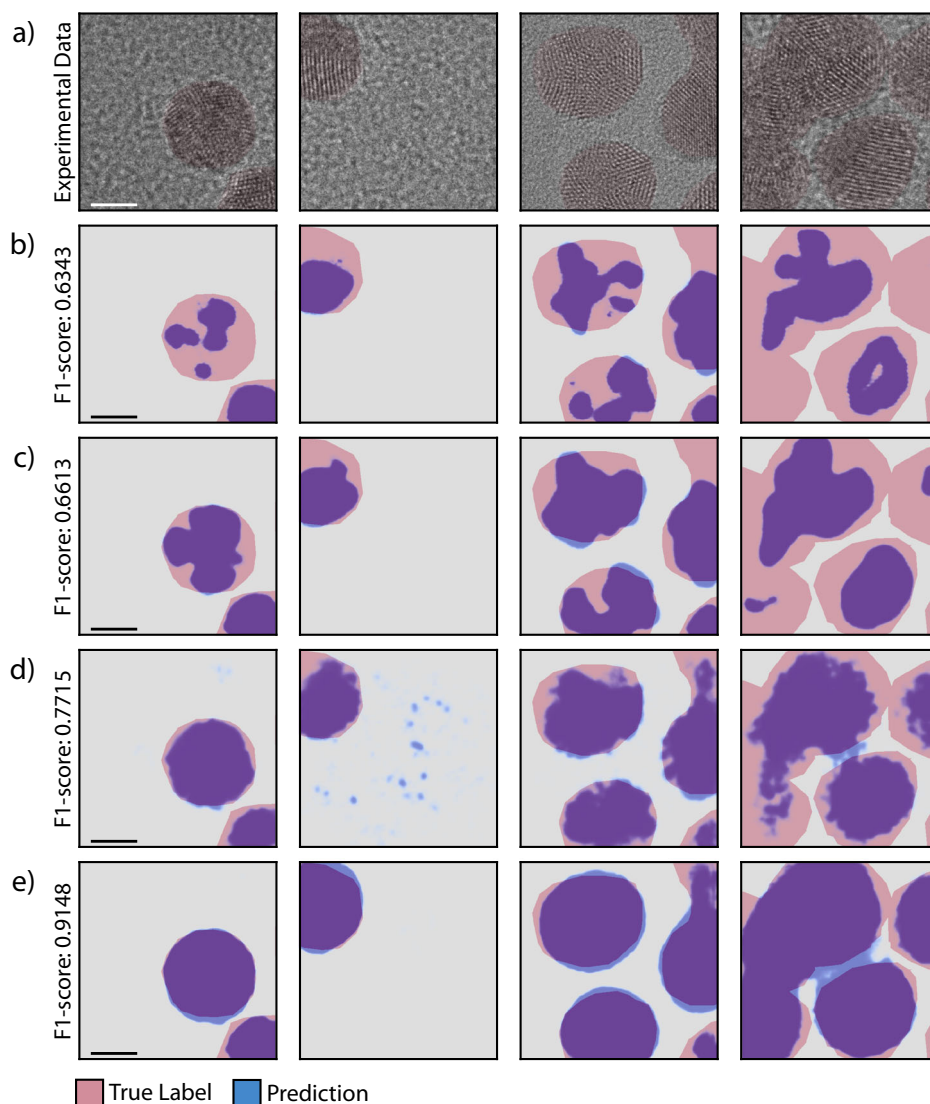
Neural networks can be trained to segment experimental HRTEM datasets with moderate accuracy even with small simulated datasets containing only 512–1024 images, as indicated by our results in Table 1. Models trained on simulated data optimize performance on the simulated training data quickly and to an extremely high degree of accuracy—frequently achieving F1-scores above 0.90 on the (simulated) validation dataset in a small number of epochs (c.f. Table 1, rightmost column)—whereas their performance on experimental data increases more slowly, continually improving throughout training, even after performance on simulated data has apparently saturated (c.f. Supplementary Fig. 13). Thus, performance on simulated data is not a reliable signal for performance on experimental data and benchmarking is crucial for successful model deployment. Model performance on validation data can lag behind training performance, reducing with increasing dataset size, similar to delays observed for models trained on small, algorithmically generated datasets²³. In Fig. 3, we visualize characteristic segmentation performance of four neural networks, across a range of accuracies, on the large Au dataset. Poorly performing segmentation models (Fig. 3b, c), after training on just simulated data, can accurately predict segmentation regions on nanoparticles with clear lattice fringes, but might miss similar particles in other micrographs and/or lose significant

performance when predicting segmentation regions for more complex structures textures and particles with many grains. Better networks have smoother, more consistent predictions (Fig. 3d, e) but still might miss regions in particles with more complex structures or might have high-frequency spatial fluctuations in their predicted regions (Fig. 3d, rightmost column), which are not physically consistent with nanoparticle structures, potentially indicating important noise features or aspects of the imaging conditions are not fully captured during data curation. Across the board, neural networks seem to segment nanoparticles more consistently when the particle (or particle grain) has visible lattice fringes, indicating that trained models can distinguish ordered lattice textures from other regions.

The overall cost of the model development process depends both on the cost of procuring effective training datasets and the cost of training the models themselves, and often, compromises must be made between acceptable costs and the end quality of obtained models. As is well known, neural networks trained with supervised learning methods can be greatly improved by using larger training datasets, which can cost much more to obtain and can cause model training time to increase substantially. In Fig. 4a, we measure the performance of neural networks on the large Au experimental dataset after training on datasets of increasing size, where the training data are drawn uniformly randomly from all of the aggregated databases (c.f. Supplementary Section 1). Segmentation performance saturates at an F1-score of about 0.9 after about 8000 images are included in the training dataset, after which there are only marginal improvements due to dataset size alone. With smaller datasets, model performance can be highly variable, and the gap between the worst and best model trained steadily decreases as the dataset size increases. By tuning the quality and composition of the simulated data, we can train comparably accurate models with more data-efficient curation strategies, as, for example, with the networks trained on structures with varying substrate thicknesses (c.f. Table 1 and, for more details, Supplementary Table 2), indicated by the gold circle in Fig. 4a.

At a fixed dataset size, the model performance can be highly sensitive and dependent on the noise of the training dataset, as shown for models trained on datasets varying only in applied electron dose in Fig. 4b. On the small Au and CdSe experimental datasets, we find that model performance maintains this sensitivity, but now peaks at lower dosages, with slow fall off of performance as noise decreases (c.f. Supplementary Figs. 9 and 10), indicating that noisier data are important for models to learn. The performance boost observed when using noisier data can potentially be seen as a regularization effect: including noisier data in the training dataset for a neural network model can help ensure that model predictions are stable with respect to noisy perturbations in experimental data. This effect is more clear when measuring the performance of our networks across all of the simulated datasets of varying noise, where the models using the lowest dose training data tend to have stable performance across all higher dosages, too (c.f. Supplementary Fig. 6). Differences in the qualitative performance trend across noise levels between the large Au data and the other experimental data might be a result of differing noise distributions in the images arising from differing camera statistics, indicating that a pure Poissonian noise model may not fully match observed noise distributions in our experimental benchmarks. Given the sensitivity of model performance to the noise level of the training dataset, we recommend to sample relevant noise from wider distributions, such that during training models see examples from a range of signal-to-noise conditions, which can improve the consistency of model training but still requires a careful choice of the noise distribution (c.f. Supplementary Fig. 2). Our results appear to be consistent as to prior work for a similar nanoparticle segmentation task²⁴ with a different experimental geometry in which nanoparticles are mounted on a crystalline substrate and are imaged over vacuum. Given that, in our task, the imaging beam passes through both the nanoparticle and the (amorphous) substrate and has higher effective electron dose (200–600 e^- per \AA^2 across all datasets), it is likely that the relationship to noise could be more complex and that the minimal experimental dose that could be consistently segmented is larger, i.e., more signal-to-noise is required for our task.

Fig. 3 | Characteristic performance of neural network models analyzing experimental images of Au nanoparticles after training on only simulated data. Models were selected trained on Baseline, Substrate, and Optimized Au datasets, from which these models represent median to strongly-performing examples. Scalebar is 2.5 nm.



In high-throughput simulation settings, instead of producing many examples over varying noise conditions, individual examples in the training dataset could be made more informative by improving the quality of the TEM simulation, which can improve the robustness of the desired dataset and shift the source of development costs. In our case, improving the fidelity of HRTEM image formation by including the effects of inelastic scattering due to thermal vibrations, losses due to plasmonic excitations, and/or residual aberrations in the optical alignment of the microscope has a significant, positive effect on neural network performance. In Fig. 5, we show the shift in performance of neural networks in the data scarce regime ($N = 512$) when trained on data with combinations of applied thermal effects, residual aberrations, and plasmonic losses as compared to a baseline dataset with no such effects. Visually, the impact of these effects can be subtle on the simulated image (Fig. 5c, d), yet, when added to the simulation data used for training, model performance can increase by as much as 0.1–0.15 in F1-score (Fig. 5b). Importantly, in a regime of stable optimization, including simulation effects appears to be helpful across a wide range of applied dosage only when all additional effects are included. Of these effects, applying thermal effects is the most computationally expensive, as it requires averaging several HRTEM wavefunctions over a set of independent frozen phonon configurations, which linearly increases simulation costs, post-processing time, and memory requirements, though, typically, only a small number of frozen phonons ($\mathcal{O}(10)$) are needed to thermally converge TEM

simulations of larger atomic structures. Applying the effects of residual aberrations and plasmonic losses are both relatively cheap in comparison, and thus, should be included in training datasets if possible.

The relationship between dataset composition—such as diversity of atomic structures or imaging conditions—to model performance is more nuanced, and in particular, aspects of dataset composition seem to be more important for controlling the variance of model performance than for boosting performance ceilings. That is, by including images of nanoparticles from a wide variety of structures or imaging conditions, we can much more likely guarantee that a randomly initialized model will optimize well and learn to segment experimental data effectively. In Fig. 6, we measure the performance of models trained on simulated datasets of fixed size ($N = 1024$) with varying atomic structural content and imaging conditions. Networks trained on datasets comprising images of samples with varying substrate thicknesses performed better and with lower variance than networks trained on a single, fixed substrate thickness (Fig. 6a, b), indicating diversity in the atomic structures seen during training is important for image segmentation. Regarding imaging conditions, relative defocus is known to have a strong nonlinear effect on contrast features in HRTEM, which is reflected in a highly variable nonlinear relationship of model performance to the focal point of simulated data (c.f. Supplementary Fig. 7). To address this, we can sample images from a larger variety of focal points (Fig. 6c, d), increasing the number of unique focal points (and by construction, focal

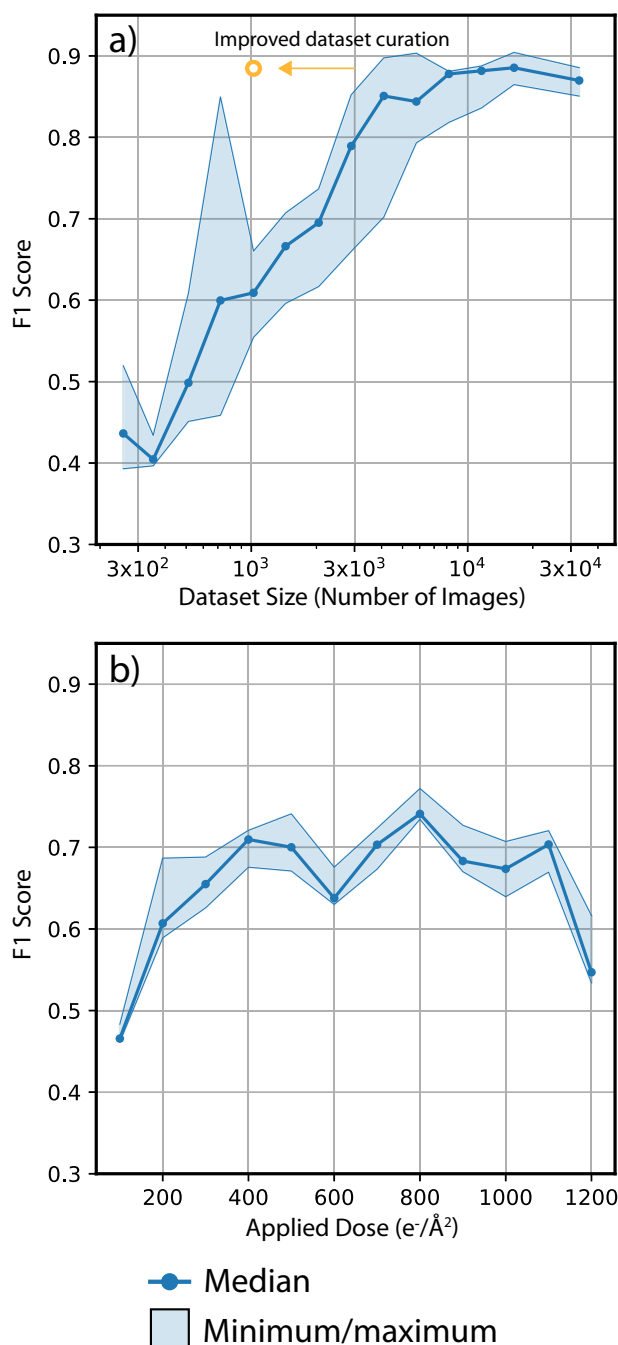


Fig. 4 | Dependence of neural network model performance on dataset size and noise. Effect of training dataset size (a) and noise from applied electron dose (b) on neural network performance, as measured on experimental data of large Au nanoparticles. Networks in (a) were trained on a uniformly random chosen subset of all simulated data in this study, excluding the ‘Optimal’ datasets. Networks in (b) were trained on data drawn from the ‘Thermal’ and ‘Dose Variation’ datasets. The gold circle indicates the performance of the best neural network trained on a dataset containing only 1024 images, which were drawn from a simulated dataset comprising structures with varying substrate thicknesses. For each dataset condition, five randomly initialized neural networks were trained and measured.

range) seen during training while maintaining the total size of the dataset, which has a small positive effect on segmentation performance and a strong effect on reducing model variance. The number of unique nanoparticle atomic structures, however, does not significantly impact model performance (Fig. 6e, f), as long as a wide variety of imaging conditions are indeed being sampled in the training dataset. In general, these data compositional effects are more significant when models are optimized more aggressively

(Fig. 6a, c, e) by using a larger initial learning rate and the extra data diversity can be interpreted as having a regularizing effect. Notably, regularization effects arising from an increase in data diversity might come into play only when the training dataset is already a suitable match to the experimental data being analyzed—i.e., that the distributions from which simulated training data and experimental data are drawn are similar. We find that these regularization effects are not noticeable when neural network performance is measured on the large Au or CdSe datasets (Supplementary Figs. 8 and 10), where performance is worse overall, possibly due to the experiments measuring nanoparticle structures that are structurally dissimilar to the simulated data used to train the networks varying in focal conditions. Changing the nanoparticle size in the training data can have a strong effect on performance (c.f. Supplementary Fig. 11), but does not appear to be completely related to the size of the nanoparticles in experimental images.

Incorporating these lessons on data effects altogether, we can design an optimized simulated dataset to target maximal absolute performance on our experimental benchmarks—ultimately, the balance of dataset size, fidelity, and composition will be dictated by the needs of a particular experiment or analytical task, which can include constraints on model size, data curation time, and model time-to-train. Here, we focused on improving performance on both the large Au dataset and the CdSe dataset by increasing the variety of atomic structures seen, including all simulation effects mentioned previously, sampling a wider range of imaging and noise conditions, and using a modestly large training dataset (8000 total images). On these benchmarks, with slight re-tuning of the training hyperparameters, we achieve a maximum F1 score of 0.9189 and 0.7516, for Au and CdSe, respectively, whilst also achieving relatively strong generalization performance (c.f. Supplementary Fig. 12). When using a random 50/50 mixture of the optimized datasets, networks have strong generalization performance across the three datasets, but better peak performance is achieved only on the small Au dataset (c.f. Table 1). For details regarding the composition of these datasets and the training strategy and performance measurements on other auxiliary datasets, please refer to Supplemental Sections 1 and 3.

Discussion

Our results, taken altogether, indicate that high-accuracy supervised models for analyzing atomic-resolution HRTEM experiments can be trained with sufficiently large, high-quality simulated databases. These synthetic datasets tend to need to be larger than what is typically curated experimentally (around 4000–8000 images), simulated with relatively high fidelity, and must contain appropriate dispersions of atomic structures, especially varying substrates, imaging conditions, particularly defocus, and applied noise to ensure consistent model performance. Critically, many observed effects of data curation strategies can be unnoticeable when networks perform poorly across the board or when the simulated dataset differs significantly from an experimental benchmark, as demonstrated by our tests varying the dataset composition at fixed dataset sizes (Fig. 6). With precise control of the prior distributions from which training datasets were drawn, we were able to isolate these successful data curation strategies for machine learning model development, which otherwise can become almost intractable due to the large observed variance of model performances under certain training dataset conditions.

The advantages of simulated data primarily arise from access to arbitrarily large datasets, a broader and controlled distribution of training data, and physics-based ground-truth measurements on the dataset. Our results indicate that, even when dealing with experimental data that arises from a highly-complex measurement process, the use of simulated data during the development of ML models can compete and even provide distinct performance gains as compared to utilizing experimental data. Alternative approaches for bridging the gap between simulated data and experimental data could involve training complementary sets of neural networks to specifically learn how to model and apply noise effects and features not captured by simulation models, e.g., with CycleGANs²⁵, though the extent to which such an approach could be as effective for HRTEM as for annular

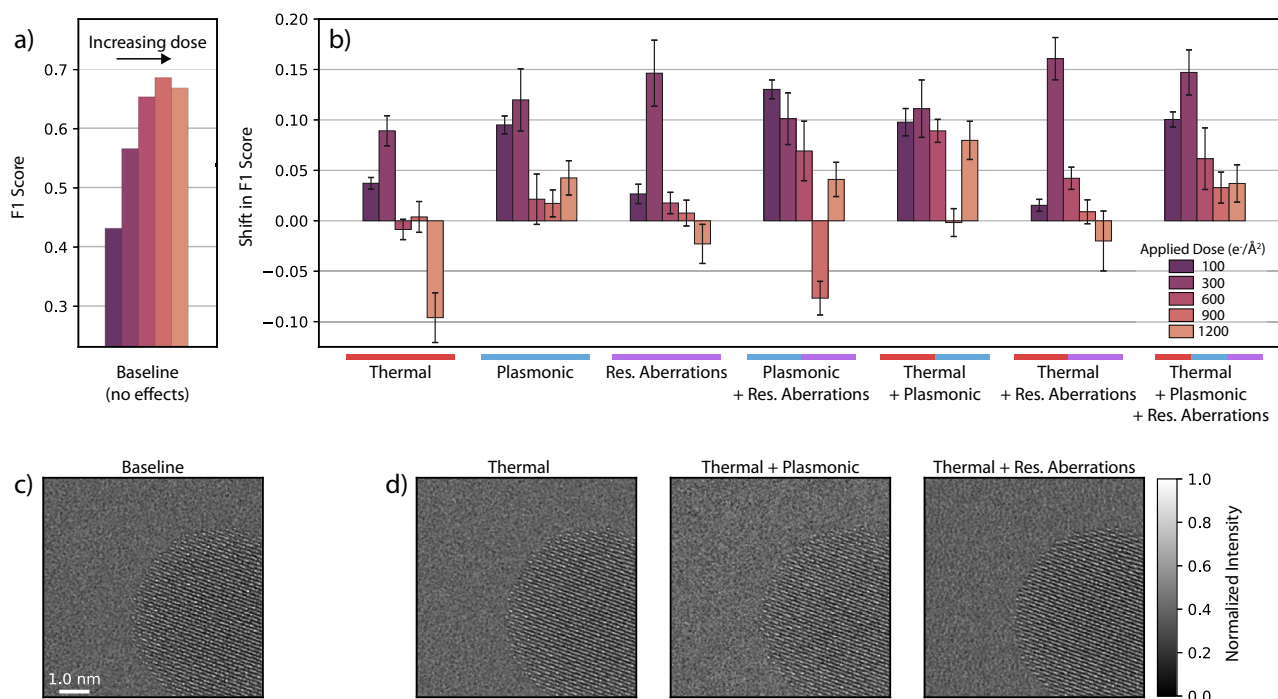


Fig. 5 | Dependence of neural network model performance on simulation fidelity. Baseline neural network performance with cheapest simulation methods (a) and effect of improving simulation fidelity on network performance, conditional on applied dose, including effects from thermal averaging, residual aberrations, and plasmonic losses (b). Visual examples of baseline quality (c) and applied simulation effects (d) on simulated image of Au nanoparticles at 300 kV with focal spread and $400 \text{ e}^-/\text{\AA}^2$. Training data for these models were drawn from the ‘Baseline’, ‘Thermal’, and ‘Simulation effects’ datasets.

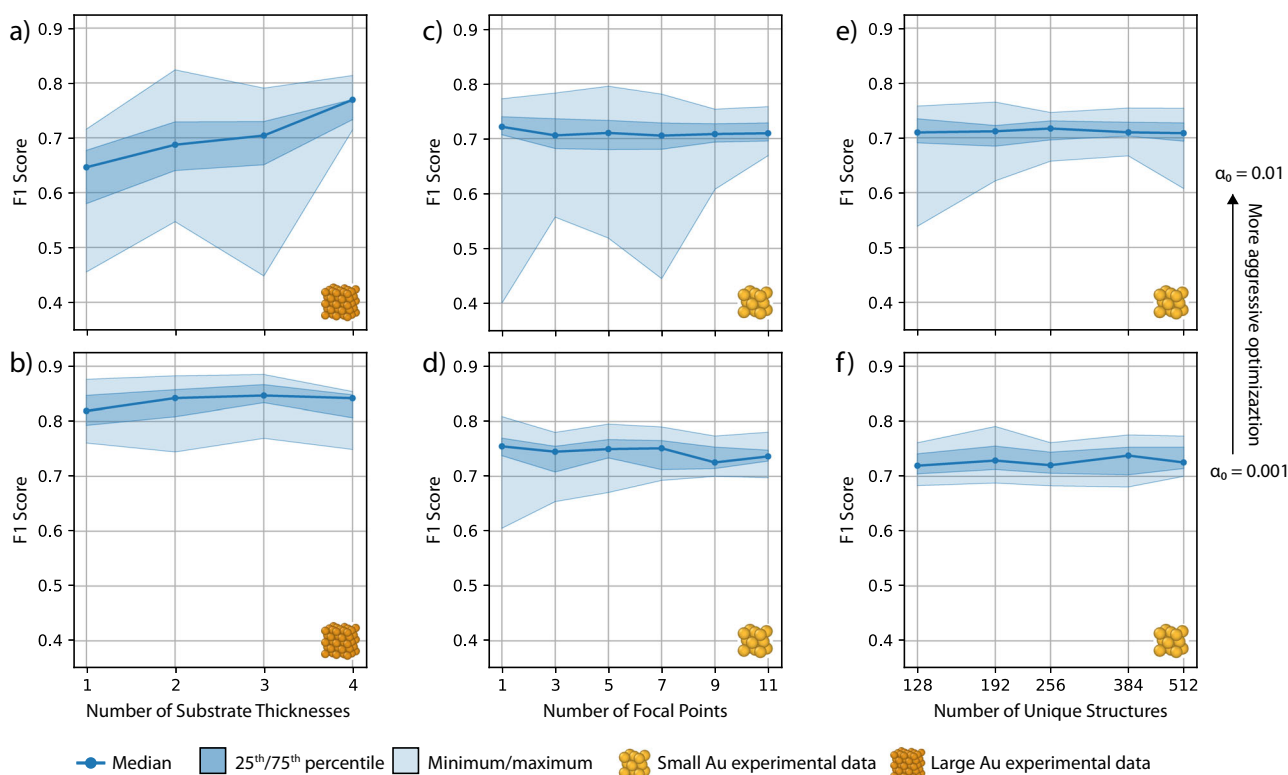


Fig. 6 | Dependence of neural network model performance on dataset composition. Effect of training dataset composition on neural network performance, at fixed noise levels and dataset size ($N = 1024$ images), varying **a**, **b** the number of substrate thicknesses of the carbon substrate in the simulated structures, **c**, **d** the number of unique focal conditions in the training dataset, and **e**, **f** the number of unique structures in the dataset. Training data for these models were drawn from the ‘Varying Substrate’ and ‘Smaller Nanoparticles’ datasets.

dark-field STEM is unclear. Well-curated experimental datasets are still immensely valuable—both as reference benchmarks, as used in this study, and as high-quality training data. An important caveat of our results is that all performance metrics shown here ultimately compare physics-based ground-truth labels and expert manual labels. Given the inevitability of human measurement error, bias, and uncertainty, the two sets of labels never match perfectly, leading to a performance gap in our models and thus an artificial performance ceiling. Further work is needed to best understand measurement error in the face of such distribution gaps and to understand the best way to develop machine learning models that can make use of such mixed data given systematically different labels.

Ideally, during training, a model would have access to a simulated dataset drawn from an identical distribution to the experimental prior and enough examples in this dataset to provide adequate coverage of the experimental distribution in which it will be deployed for analysis. In this setting, ML models only make in-distribution predictions, and never face extrapolation errors. In practice, it can be difficult to determine the exact experimental prior, suggesting instead a strategy of drawing training data from a distribution which the experimental distribution could be a feasible subset, thus, reducing adverse effects of distributional shifts during model deployment. General-purpose tools like Construction Zone, which enable fast sampling of realistic synthetic examples for ML problems and are not limited to any specific domain, are a crucial component of robust data workflows which would enable such sampling strategies. Further, such general purpose tools enable careful tuning of training datasets to best match experimental prior distributions, which can be performed manually or automatically, for example, as a component in an active learning training loop^{26,27}. Curating the best suitable dataset for a particular ML problem is a crucial component of reliable ML workflows, but should be pursued in conjunction with finding the best model framework, which can include developing bespoke model architectures^{28,29} or implementing custom loss functions³⁰ with regularization³¹ specific to the problem of interest.

In sum, we have developed Construction Zone, a Python package that enables programmatic generation and sampling of atomic nanostructures, a general purpose tool designed primarily designed to help study and design ML workflows at scale for nanoscale materials science problems. In conjunction with HRTEM simulation, we systematically generate large structural and imaging databases for training ML models and have demonstrated their utility for atomic-scale characterization with HRTEM. By systematically generating large structural and imaging databases, we achieve state-of-the-art performance on such models, whilst also providing the ability to design and develop ML tools carefully through high levels of specific control on the data generation process.

Methods

Construction Zone

Nanoscale atomic structures can be designed in Construction Zone (CZ) available at https://github.com/lerandc/construction_zone using a combination of Generator, Volume, Transformation, and Scene objects. Generators, which supply atoms in space, specify atom positions and can generate atoms in both crystalline and non-crystalline arrangements. Volume objects define the convex regions in which atoms from Generators are accepted. Volumes can either be defined by sets of convex algebraic surfaces, convex hulls of point clouds, or intersections of supplied convex objects; non-convex geometries can be created as the union of multiple Volumes. The Transformation module manipulates objects with standard routines like symmetry operations and more complex routines like applying inhomogeneous strain fields or modifying the local chemistry of a structure. Generators and Volumes can be manipulated jointly or individually for full construction flexibility. For example, a grain can be oriented either by rotating a Generator lattice relative to its boundary or by rotating the whole object in global coordinates. Finally, Scenes aggregate objects together. Given a set of atomic objects, a Scene will handle interactions through a generation precedence scheme and prepares data for interfacing with other simulation methods or for writing to file.

CZ also features a small set of auxiliary modules for more complex functionality and convenience. The Surface module provides fast routines for analyzing, querying, and modifying the surface of a generated object using derivatives of alpha-shape algorithms^{32,33}. The Utilities module provides several analysis routines, such as radial distribution function (RDF) analysis and orientation sampling. Lastly, the Prefab module contains routines that generate “pre-packaged” objects like Wulff constructions of nanoparticles (as implemented in ref. 15) or grain structures with planar defects. For a full discussion of underlying routines, available features, and package usage, we refer the reader to the CZ documentation, available at <https://construction-zone.readthedocs.io>.

TEM simulation

High-resolution TEM image simulations in this study were performed using the multislice algorithm¹⁸, as implemented in the Prismatic software package⁸. In the multislice algorithm, we model the image created in an electron microscope through the interaction of an electron beam and a material sample as the evolution of a 2D complex wavefunction, $\psi(\mathbf{r})$. The evolution of the wavefunction is given by the Schrödinger equation for fast electrons³⁴

$$\frac{\partial \psi(\mathbf{r})}{\partial z} = \frac{i\lambda}{4\pi} \nabla^2 \psi(\mathbf{r}) + i\sigma V(\mathbf{r})\psi(\mathbf{r}), \quad (1)$$

where $V(\mathbf{r})$ is the electrostatic potential of the sample and σ is an interaction constant. $V(\mathbf{r})$ is typically calculated with an isolated atom approach, where the total potential is the sum of independent atomic potentials

$$V(\mathbf{r}) = \sum_i V_i(\mathbf{r}) \quad (2)$$

where the individual atomic potentials themselves calculated with a parameterized look-up table of electron scattering factors^{35,36}. Alternatively, the scattering potential can be determined for a sample through ab initio techniques¹¹. Once $V(\mathbf{r})$ is determined, it is split into a series of binned slices along the beam direction and then the wavefunction $\psi(\mathbf{r})$ is evolved through a split-step method, where the electron beam alternately interacts with the sample

$$\psi(\mathbf{r}) = T(\mathbf{r})\psi_0(\mathbf{r}) = e^{i\sigma V_n(\mathbf{r})}\psi_0(\mathbf{r}), \quad (3)$$

and then propagates in free space

$$\psi(\mathbf{r}) = \hat{\mathcal{F}}_{\mathbf{k} \rightarrow \mathbf{r}}^{-1} \left\{ e^{i\lambda \Delta z |\mathbf{k}|^2} \hat{\mathcal{F}}_{\mathbf{r} \rightarrow \mathbf{k}} [\psi_0(\mathbf{r})] \right\}. \quad (4)$$

After the wavefunction has propagated through the entire sample, we obtain the exit wavefunction, which can be further modified to apply defocus and other residual optical aberrations typically seen in HRTEM imaging by applying another transmission operation

$$\Psi(\mathbf{k}) = \Psi_0(\mathbf{k})e^{-i\chi(\mathbf{k})} \quad (5)$$

where $\Psi_0(\mathbf{k})$ is the unaberrated wavefunction in Fourier space and χ is the aberration function. Aberration functions used to model imaging conditions in our study are detailed in Supplementary Table 1 and Supplemental Section 1.

Database generation with CZ and HRTEM simulation

We simulate a HRTEM data stream by first generating a large database of semi-realistic structures and then simulating micrographs for those structures under suitable imaging conditions. We greatly approximate the variety of nanoparticles imaged in ref. 16 as a series of spherical Au nanoparticles of varying diameters with a small number of planar defects, which are equally likely to be twin defects or stacking faults. Each nanoparticle is placed upon a

unique amorphous carbon substrate and at a random location on the substrate surface at a random orientation. We then perform plane-wave multislice simulations of the structures at an acceleration voltage of 300 kV, focused to the center of the atomic model, at a resolution of 0.02 nm per pixel. We simulate output wavefunctions for both the full structure and just the nanoparticle in vacuum. To create ground-truth segmentation masks, we threshold the phase of the output wavefunctions from the nanoparticles in vacuum, averaged over frozen phonons. Threshold values were determined heuristically. We found phase-thresholding to be much more stable and physically consistent across focal conditions than analogous intensity thresholding techniques.

With the output wavefunctions, we then apply relevant focal conditions, residual aberration effects, focal spread, thermal effects, plasmonic losses, and electron dosage effects. Focal point, focal spread, and residual aberrations are applied to output wavefunctions using Eq. (5) with the appropriate aberration function; the effect of focal spread is approximated by incoherently averaging the wavefunction over a series of focal points distributed about the intended focal point. Thermal effects are included via the frozen-phonon approximation, where we incoherently average several exit wavefunctions to approximate the effect of thermal motion, corresponding to unique, perturbed copies of the original atomic structure. The atomic positions are perturbed by I.I.D. Gaussian noise, with amplitudes given by atomic Debye-Waller factors; for all simulations including thermal effects, we used eight frozen phonons. We approximate plasmonic losses in HRTEM imaging by applying a contrast reduction to the wavefunction intensities before averaging over frozen phonons. The contrast reduction is applied by taking a weighted pixelwise mean of the original intensity and an intensity with a constant added background (renormalized to have its original mean value). Noise from applied electron dosage is modeled by sampling count intensities from scaled Poisson distributions. We note that we do not include potential artifacts from the camera, i.e. via the modulation transfer function, though all experimental datasets have been acquired by scintillator-based detectors, for which including effects from the modulation transfer function in the noise sampling process could further improve performance^{6,24}. For more details regarding the composition and sampling of specific datasets, we refer the reader to Supplemental Section 1.

Neural network training

For our neural network development, we used a standard UNet architecture^{16,19,37} with a ResNet 18³⁸ encoder/decoder architecture and three pooling/upsampling stages, resulting in about 14 million trainable parameters (c.f. Supplementary Fig. 4). Models were trained using a mixed categorical cross-entropy and F1-score loss function on the segmentation masks. We performed a brief manual hyperparameter search to determine initial learning rates and learning rate schedules which were suitable enough for stable optimization under a wide variety of the simulated datasets. For most networks whose performance is reported in this study, we used the Adam optimization algorithm³⁹ with initial learning rates of 0.01 and 0.001, a constant learning rate decay of 0.8, and a batch size of 16 images; for more detailed effects of the batch size on training dynamics, we refer the reader to Supplementary Fig. 5. Unless explicitly indicated otherwise, results in the primary text reflect those of networks trained with an initial learning rate of 0.001. Models were trained for a fixed number of epochs (25) with no early stopping; the network parameters that achieved the lowest validation loss were saved via checkpointing. Input data were first normalized, per image, to a range of 0 to 1, and then augmented with orthogonal rotations and random left-right/up-down flips. For experimental data, images were processed with a 3 × 3 median filter and then normalized to a range of 0 to 1, per image, before evaluating neural network test performance. Experimental image patches containing only substrate regions were removed from the training dataset prior to preprocessing. Per simulated training data condition tested, at least five different networks were trained with different random weight initializations and training data subsets (drawn uniformly at random after appropriate filtering). For each network, relevant metadata were saved alongside the training history and model parameters. All training

Table 2 | Performance timings for structure generation, HRTEM simulation, image generation, model training, and model inference using a single workstation and, where timings available, on a single CPU node on the Perlmutter supercomputer at NERSC

Task	Compute cost (Workstation)	Compute cost (Perlmutter)
Structure generation	15.2s/structure pair	4s/structure pair
HRTEM simulation	1.125s/frozen phonon	0.294s/frozen phonon
Image generation	1.4s/image (G)	0.108s/image
Model training	15s/512 images/epoch (G)	—
Model inference	0.014s/512 × 512 patch (G)	—

Where relevant, GPU usage is indicated with the marking (G). Perlmutter timings utilize 32 processes with 8 threads each.

was performed using Tensorflow version 2⁴⁰, with the Segmentation Models package⁴¹. Neural networks were benchmarked for performance on three experimental datasets using the F1-score

$$F1 = \frac{2f(x) \cdot \hat{y}}{\|f(x)\| + \|\hat{y}\|} \quad (6)$$

where f is the neural network model, x is the input image, and \hat{y} is the corresponding image label. When discretizing the prediction to binary values, the F1 score can be interpreted as a ratio of true positives (identifying nanoparticle region correctly) to a sum of true positives, false positives (identifying substrate as nanoparticle) and false negatives (identifying nanoparticle as substrate) such that $F1 = \frac{TP}{TP+0.5(FN+FP)}$. The F1 score measures 0 when all pixels are misclassified and 1 only when there are zero misclassified pixels. The non-discretized form is used for training and evaluation in this work.

Computational resources and benchmarks

All structure generation, simulation, data generation, and machine learning were performed on a workstation computer, using an Intel Xeon Gold 6130 CPU (16 cores, 256 Gb RAM) and a NVIDIA Quadro P5000 GPU (16 Gb RAM). Table 2 shows representative benchmark timings for each of the main compute tasks in our training pipeline, as logged by our data generation scripts. To understand how our generation pipeline scales with typical modern high performance computing resources, we ran comparable data generation tasks on the Perlmutter supercomputer at NERSC, using a single CPU node comprising 2 AMD EPYC 7763 CPUs (64 cores ea., 512 Gb RAM total). Our database generation workflow can generally be run in an embarrassingly parallel fashion, where multiple processes are spawned to generate larger databases by, e.g., individually handling independent structures, simulations, or images; the timings below represent the time per item when running in parallel. In our pipeline, filesystem performance can heavily influence the computational cost since each intermediate output was saved to disk for reproducibility, reducing the effective parallelism of the approach, especially on the workstation. We suspect time-to-compute, from specification of a structure to a set of corresponding training images, can be reduced significantly by instead keeping things in working memory. The structure generation costs are dominated by the generation of the amorphous carbon substrate, which is both serial and difficult to parallelize, but could be precomputed. The HRTEM simulation and image generation tasks are both dominated by the fast Fourier transform (FFT) operations, in the propagation of the electron wavefunction and the application of focal spread, respectively, and thus greatly benefit from having an optimized shared-memory FFT implementation.

Data availability

The original atomic structures and the processed images used to train segmentation models in this study have been made available via Zenodo at

<https://zenodo.org/doi/10.5281/zenodo.11554183>; we also include a series of trained model weights, from models trained on the baseline, substrate varying, and optimized datasets.

Code availability

Code used to generate atomic structures, HRTEM simulations, and fully processed images, as well as code used to train neural networks models, has been made available at <https://github.com/ScottLabUCB/HRTEM-robust-synthetic-data>. The Construction Zone package is available on Github at https://github.com/lerandc/construction_zone, with documentation available at <https://construction-zone.readthedocs.io/>.

Received: 22 August 2023; Accepted: 4 July 2024;

Published online: 29 July 2024

References

- Xu, K. et al. How Neural networks Extrapolate: From Feedforward to Graph Neural Networks. In *Proceedings of the 9th International Conference on Learning Representations*. <https://openreview.net/forum?id=UH-cmoclJC> (2021).
- Nagarajan, V., Andreassen, A. & Neyshabur, B. Understanding the failure modes of out-of-distribution generalization. In *Proceedings of the 9th International Conference on Learning Representations*. https://openreview.net/forum?id=fSTD6NFIW_b (2021).
- Krueger, D. et al. Out-of-distribution generalization via risk extrapolation (REx). In *Proceedings of the 38th International Conference on Machine Learning*, 5815–5826 (2021).
- Munshi, J. et al. Disentangling multiple scattering with deep learning: application to strain mapping from electron diffraction patterns. *npj Comput. Mater.* **8**, 1–15 (2022).
- Zhang, C., Feng, J., DaCosta, L. R. & Voyles, P. M. Atomic resolution convergent beam electron diffraction analysis using convolutional neural networks. *Ultramicroscopy* **210**, 112921 (2020).
- Madsen, J. et al. A deep learning approach to identify local structures in atomic-resolution transmission electron microscopy images. *Adv. Theory Simul.* **1**, 1800037 (2018).
- Vincent, J. L. et al. Developing and evaluating deep neural network-based denoising for nanoparticle TEM images with ultra-low signal-to-noise. *Microsc. Microanal.* **27**, 1431–1447 (2021).
- Rangel DaCosta, L. et al. Prismatic 2.0—simulation software for scanning and high resolution transmission electron microscopy (STEM and HRTEM). *Micron* **151**, 103141 (2021).
- Pryor, A., Ophus, C. & Miao, J. A streaming multi-GPU implementation of image simulation algorithms for scanning transmission electron microscopy. *Adv. Struct. Chem. Imaging* **3**, 15 (2017).
- Ophus, C. A fast image simulation algorithm for scanning transmission electron microscopy. *Adv. Struct. Chem. Imag.* **3**, 13 (2017).
- Madsen, J. & Susi, T. The abTEM code: transmission electron microscopy from first principles. *Open Res. Eur.* **1**, 13015 (2021).
- Lobato, I. & Van Dyck, D. MULTTEM: a new multislice program to perform accurate and fast electron diffraction and imaging simulations using Graphics Processing Units with CUDA. *Ultramicroscopy* **156**, 9–17 (2015).
- Ong, S. P. et al. Python Materials Genomics (pymatgen): a robust, open-source Python library for materials analysis. *Comput. Mater. Sci.* **68**, 314–319 (2013).
- Larsen, A. H. et al. The atomic simulation environment—a Python library for working with atoms. *J. Phys.: Condens. Matter* **29**, 273002 (2017).
- Rahm, J. M. & Erhart, P. WulffPack: a Python package for Wulff constructions. *J. Open Source Softw.* **5**, 1944 (2020).
- Groschner, C. K., Choi, C. & Scott, M. C. Machine learning pipeline for segmentation and defect identification from high-resolution transmission electron microscopy data. *Microsc. Microanal.* **27**, 549–556 (2021).
- Wang, X. et al. AutoDetect-mNP: an unsupervised machine learning algorithm for automated analysis of transmission electron microscope images of metal nanoparticles. *JACS Au* **1**, 316–327 (2021).
- Cowley, J. M. & Moodie, A. F. The scattering of electrons by atoms and crystals. I. A new theoretical approach. *Acta Crystallogr* **10**, 609–619 (1957).
- Ronneberger, O., Fischer, P. & Brox, T. U-Net: convolutional networks for biomedical image segmentation. Preprint at <https://arxiv.org/abs/1505.04597> (2015).
- Sytwu, K., Groschner, C. & Scott, M. C. Understanding the influence of receptive field and network complexity in neural network-guided TEM image analysis. *Microsc. Microanal.* 1–9 (2022).
- Groschner, C., Choi, C. & Scott, M. C. High throughput pipeline for segmentation and defect identification. <https://doi.org/10.5281/zenodo.3755011> (2020).
- Sytwu, K., Groschner, C. & Scott, M. C. Understanding the influence of receptive field and network complexity in neural-network-guided TEM image analysis. <https://doi.org/10.5281/zenodo.6419024> (2022).
- Power, A., Burda, Y., Edwards, H., Babuschkin, I. & Misra, V. Grokking: generalization beyond overfitting on small algorithmic datasets. Preprint at <https://arxiv.org/abs/2201.02177> (2022).
- Leth Larsen, M. H., Lomholdt, W. B., Nuñez Valencia, C., Hansen, T. W. & Schiøtz, J. Quantifying noise limitations of neural network segmentations in high-resolution transmission electron microscopy. *Ultramicroscopy* **253**, 113803 (2023).
- Khan, A., Lee, C.-H., Huang, P. Y. & Clark, B. K. Leveraging generative adversarial networks to create realistic scanning transmission electron microscopy images. *npj Comput. Mater.* **9**, 1–9 (2023).
- Ang, S. J., Wang, W., Schwalbe-Koda, D., Axelrod, S. & Gómez-Bombarelli, R. Active learning accelerates ab initio molecular dynamics on reactive energy surfaces. *Chem* **7**, 738–751 (2021).
- Vandermaese, J. et al. On-the-fly active learning of interpretable Bayesian force fields for atomistic rare events. *npj Comput. Mater.* **6**, 1–11 (2020).
- Xie, T. & Grossman, J. C. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Phys. Rev. Lett.* **120**, 145301 (2018).
- Musaelian, A. et al. Learning local equivariant representations for large-scale atomistic dynamics. *Nat. Commun.* **14**, 579 (2023).
- Kervadec, H. et al. Boundary loss for highly unbalanced segmentation. *Med. Image Anal.* **67**, 101851 (2021).
- Kirkpatrick, J. et al. Pushing the frontiers of density functionals by solving the fractional electron problem. *Science* **374**, 1385–1389 (2021).
- Stukowski, A. Computational analysis methods in atomistic modeling of crystals. *JOM* **66**, 399–407 (2014).
- Edelsbrunner, H. & Mücke, E. P. Three-dimensional alpha shapes. *ACM Trans. Graph.* **13**, 43–72 (1994).
- Van Dyck, D. *Advances in Electronics and Electron Physics* 295 (Academic, 1985).
- Kirkland, E. J. *Advanced Computing in Electron Microscopy* (Springer US, 2010).
- Lobato, I. & Van Dyck, D. An accurate parameterization for scattering factors, electron densities and electrostatic potentials for neutral atoms that obey all physical constraints. *Acta Crystallogr A* **70**, 636–649 (2014).
- Zhou, Z., Siddiquee, M. M. R., Tajbakhsh, N. & Liang, J. UNet++: redesigning skip connections to exploit multiscale features in image segmentation. Preprint at <https://arxiv.org/abs/1912.05074> (2020).
- He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778 (2016).
- Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. Preprint at <https://arxiv.org/abs/1505.0459> (2017).

40. Abadi, M. et al. TensorFlow: A system for large-scale machine learning. In *12th USENIX Symposium on Operating Systems Design and Implementation, OSDI 2016, Savannah, GA, USA, November 2–4, 2016* (eds. Keeton, K. & Roscoe, T.) 265–283 (USENIX Association, 2016).
41. Iakubovskii, P. Segmentation models, https://github.com/qubvel/segmentation_models, (2019).
42. Oh, M. H. et al. Design and synthesis of multigrain nanocrystals via geometric misfit strain. *Nature* **577**, 359–363 (2020).
43. Reidy, K. et al. Direct imaging and electronic structure modulation of moiré superlattices at the 2D/3D interface. *Nat. Commun.* **12**, 1290 (2021).

Acknowledgements

The authors thank Dr. Colin Ophus and Dr. Philipp Pelz for helpful discussions. This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, Department of Energy Computational Science Graduate Fellowship under Award Number DE-SC0021110. K.S. was supported by an appointment to the Intelligence Community Postdoctoral Research Fellowship Program at Lawrence Berkeley National Laboratory administered by Oak Ridge Institute for Science and Education (ORISE) through an interagency agreement between the U.S. Department of Energy and the Office of the Director of National Intelligence (ODNI). This work was also funded by the US Department of Energy in the program “4D Camera Distillery: From Massive Electron Microscopy Scattering Data to Useful Information with AI/ML.” Work at the Molecular Foundry was supported by the Office of Science, Office of Basic Energy Sciences, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using NERSC award BES-ERCAP0026467. This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views

and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Author contributions

L.R.D. performed the simulated data generation, machine learning, and data analysis and prepared the manuscript. K.S. and C.K.G. acquired and labeled the experimental data, provided assistance with the machine learning, and helped prepare the manuscript; C.K.G. also developed initial workflows for the machine learning. M.C.S. led and supervised the study and helped prepare the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41524-024-01336-0>.

Correspondence and requests for materials should be addressed to Luis Rangel DaCosta or M. C. Scott.

Reprints and permissions information is available at <http://www.nature.com/reprints>

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article’s Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article’s Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2024