Automated Continual Learning of Defect Identification in Coherent Diffraction Imaging

Orcun Yildiz*, Henry Chan*, Krishnan Raghavan*, William Judge†, Mathew J. Cherukara*, Prasanna Balaprakash*, Subramanian Sankaranarayanan*, and Tom Peterka*
*Argonne National Laboratory, IL 60439, USA, {oyildiz, hchan, kraghavan, mcherukara, pbalapra, ssankaranarayanan, tpeterka}@anl.gov
†Formerly of Department of Chemistry, University of Illinois, Chicago, IL 60607, USA, {wjudge2}@uic.edu

***Abstract*—X-ray Bragg coherent diffraction imaging (BCDI) is widely used for materials characterization. However, obtaining X-ray diffraction data is difficult and computationally intensive. Here, we introduce a machine learning approach to identify crystalline line defects in samples from the raw coherent diffraction data. To automate this process, we compose a workflow coupling coherent diffraction data generation with training and inference of deep neural network defect classifiers. In particular, we adopt a continual learning approach, where we generate training and inference data as needed based on the accuracy of the defect classifier instead of all training data generated a priori. The results show that our approach improves the accuracy of defect classifiers while using much fewer samples of data.**

***Index Terms*—HPC workflows, defect identification, continual learning, catastrophic forgetting**

I. Introduction

The properties and performance of materials are strongly influenced by the presence of defects. For instance, mechanical hardness of some metals can be increased through work hardening, a plastic deformation process that increases the number of dislocation defects arise from the sliding of atoms from their ideal positions in the crystal structure. The identification and precise control of defects in materials not only plays an important role in the fundamental understanding of novel materials, but also in advanced manufacturing applications such as fault detection and strain engineering.

Traditionally, defect identification from images of a material sample is performed using computer vision and statistical approaches, which involves procedures such as image segmentation and pattern/feature extraction. With the advent of high-resolution/high-throughput instruments, such as the fourth-generation synchrotron X-ray sources, machine learning based methods such as Deep Convolutional Neural Networks (DNN) are increasingly used to automate and accelerate the analysis of big data streams. In the context of diffractive imaging, min-max optimization [1], guided algorithms [2], and DNN [3] have been used for the identification of dislocation defects in nanocrystals.

While neural networks can improve the speed and accuracy of defect identification, they require generating large amounts of training data, which is a computationally and manually intensive task. Moreover, the full science pipeline of defect identification requires the combination of different types of tasks such as crystal data generation, full simulations, and artificial intelligence models. Managing all these tasks can be burdensome for scientists; hence, seamless and automatic management of these tasks is required.

Motivated by these needs, in this work we introduce an automated workflow for defect identification that dynamically generates training data on demand as a neural network is being trained. Because obtaining data from physical light source experiments is time-consuming and expensive, our goal is to minimize the amount of data needed to train the neural network. In particular, we adopt a continual learning approach, which can incrementally train the model when the data are observed sequentially rather than in bulk. We summarize our contributions as follows:

- We develop an automated workflow using online crystal generation coupled with training and inference of deep neural network defect classifiers for automating defect classification in coherent diffraction imaging.
- We adopt a continual learning approach for improved accuracy of defect classifiers when presented with training data sequentially.
- We compare our continual learning (CL) approach against single-shot learning (SSL) and naive incremental learning (NIL) approaches. In SSL, all of the training data are generated a priori. NIL updates the model naively with sequential data, but suffers from catastrophic forgetting. The CL approach achieves higher accuracy than NIL while using much less training data than SSL.

The remainder of this paper is organized as follows. Section II presents background and related work. Section III describes our workflow methodology for automating defect classification in coherent diffraction imaging. Section IV presents our experiment design, followed by experimental results in Section V. Section VI concludes the paper with a summary and a brief look at future work.

II. Background and Related Work

In this section, we provide brief background and related work on defect identification, workflow management, and continual learning.

A. Defect Identification in Coherent Diffraction Imaging

Bragg coherent X-ray diffraction imaging (BCDI) is a technique that can probe the size, shape, strain, and spatial

location of defects in crystalline nanomaterials. In BCDI, a nanomaterial sample is studied using scattered intensities generated by a coherent X-ray beam measured at the Bragg peak. By collecting the scattered intensities along a rocking curve, a 3D diffraction pattern can be acquired. The diffraction pattern is the modulus of the complex Fourier transform (FT) of the sample without direct information about phases of the scattered waves. Therefore, the amplitude and phase of the nanomaterial sample can only be reconstructed using a phase-retrieval algorithm [4]. Since these algorithms are iterative in nature, neural network models have been applied to overcome the limitation on speed and accuracy [5]–[10]. Furthermore, these machine learning approaches have the potential to fully reconstruct samples with defects that are challenging to solve using the traditional approaches.

There exist several research efforts on using machine learning techniques for identifying defects in coherent diffraction imaging [1]–[3]. Ulvestad et al. have developed a min-max optimization approach for extracting the position of dislocation lines in reconstructed BCDI images [1]. They have also outlined a general prescription for phasing of nanocrystals with defects, based on studying simulated diffraction images using guided algorithms [2]. Recently, Bruce et al. trained a neural network model on synthetic data for the identification of dislocation defects and validated the model on experimental data [3]. Unlike these studies that require large datasets, in this work we aim to perform defect identification by using smaller datasets.

B. Workflow management

Scientific computing consists of multiple related computational tasks. Scientific workflow frameworks allow scientists to define the dependencies and data exchanges among connected tasks instead of managing those manually, potentially resulting in increased scientific productivity. Workflows are often characterized in two types—distributed (cloud) workflows or in situ (HPC) workflows. Representative examples of distributed workflows, where tasks can run across several independent systems in a wide area such as grids and clouds, include recent publications by Deelman et al. [11] and Altintas et al. [12]. On the other hand, in situ workflows run within a single high-performance computing (HPC) system, and launch all tasks concurrently. Examples include ParaView Catalyst [13], VisIt Libsim [14], ADIOS [15], SENSEI [16], and Damaris [17].

In this paper, we use the Decaf [18] in situ workflow tool for performing defect identification in an automated fashion. Decaf allows parallel communication of coupled tasks by creating communication channels over HPC interconnects through MPI. It provides a Python API to describe the workflow task graph. Decaf does not impose any constraints on this graph topology and can manage graphs with cycles as in our defect identification workflow, where we generate training data on demand while a neural network is being trained.

C. Continual learning

Automated identification of defects in diffraction imaging requires a considerable amount of data. However, generating data through real experiments is often expensive and impractical. Nevertheless, even when it is possible to generate such data, these data are not available altogether and are typically generated on demand. However, training machine learning (ML) on sequentially generated data presents a quandary. When an ML model is presented with sequentially generated data, it exhibits a phenomenon known as catastrophic forgetting [19], [20]—where a model forgets (overwrites) previously learned information when encountering new information.

Continual learning (CL) is an important paradigm that attempts to minimize catastrophic forgetting while efficiently learning on new sequential data (generalization). In other words, CL seeks a balance between minimizing forgetting and improving generalization. CL has been extensively studied in recent years [19]–[22]. While earlier works in [19], [20] focus on improving either generalization or forgetting, prior work in [21] introduces a two-player game that seeks to find a balance between generalization and forgetting with theoretical guarantees. In this work, we employ the CL approach introduced in [21] in order to identify defects in coherent diffraction imaging with sequential data generation. In particular, we try to learn on the newly available defect data while minimizing the forgetting on already learned information from the previous defect data.

III. Methodology

In this section, we present an overview of the defect classification workflow and describe its main components.

A. Automating the defect classification workflow

In this work, we generate digital twins of simulated crystals as proxies for physical experiments. As in actual experiments, we aim to minimize the number of crystals generated to train the DNN, which is what motivates our defect classification workflow.

To automate the defect classification, we couple coherent diffraction data generation with training and inference of deep neural network defect classifiers. Figure 1 displays this end-to-end workflow, which is orchestrated by the Decaf workflow system. This workflow consists of four main steps. First, crystals with three different defect types (edge-type, screw-type, and defect-free) are generated using AtomSK [23], a software for creating crystal structures for atomic-scale simulations. Second, generated crystal structures are energetically relaxed using LAMMPS [24], a parallel molecular dynamics (MD) simulation package. The atomic configurations with and without energy minimized are then used to simulate BCDI diffraction patterns with the PyNX [25] scattering module. Finally, we feed these diffraction data into our continual learning model using PyTorch library [26] for training and inference. This workflow iterates, continually generating training data and updating the DNN until the trained model converges.

The challenges in orchestrating this workflow arise from the heterogeneity of the constituent tasks in terms of their resource requirements, programming, and data models. For instance, AtomSK is a serial Fortran program, while LAMMPS is a

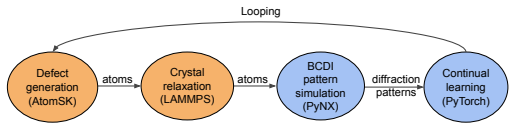


Fig. 1. Automated workflow of continual learning for defect classification.

parallel MPI program written in C++. Moreover, diffraction pattern generation and model training and inference tasks (shown in blue in Figure 1) require GPU resources, while remaining tasks of the workflow (shown in yellow in Figure 1) run on CPUs. Employing tasks with heterogeneous requirements is quite common in today's scientific applications, and our workflow enables development of such applications and automates their execution.

Next, we discuss the details of the data generation and continual learning approach.

B. Data generation

Our data generation process employs the methodology used by Chan et al. [9]. For the CL and NIL approaches, we generate data online sequentially, on demand. For SSL, we generate all the data offline in bulk, in advance. Table I summarizes the amount of the data for each stage of both online sequential and offline bulk data generation.

Synthetic data of defective gold nanocrystals are generated using atomistic simulations. The AtomSK software [27] is used to create initial crystals of different shapes from a $\approx$ 20 x 20 x 20 nm^3 face-centered cubic lattice of gold atoms. These crystals have varying sizes, number of facets, and crystal orientations. Different defect types (edge, screw, and defect-free) are introduced to each crystal and subsequently relaxed in molecular dynamics simulations using the LAMMPS software [28].

Simulated diffraction patterns are created from the atomistic gold nanocrystals using the PyNX software [29]. The 3-dimensional Bragg coherent diffraction patterns of size 128^3 pixels are simulated around several reflections of the crystals. Bragg peaks that are insensitive to the presence of one or more defects, meeting the invisibility condition [30], are excluded. This protocol keeps the same distribution for classification across defect types.

2-dimensional slices of size 128^2 pixels are extracted from the 3-dimensional Bragg coherent diffraction patterns. To obtain all central slices around a given Bragg peak, each BCDI pattern is rotated in the 24 unique starting orientations of a cube. At each orientation, slices are taken at every 5° of the [-45°, +45°] rotation interval. This slicing procedure ensures the dataset contains all central slices for any given nanocrystal orientation, and the slices are simulated deviations from a perfect (100)(010)(001) crystal orientation.

C. Continual learning

In our workflow, we are given a new set of crystal images at every iteration k. Our goal is to learn to predict whether these images present any defects and classify their type. Let $\boldsymbol{\mathcal{Q}}_k$ represent this task at instant k such that $\boldsymbol{\mathcal{Q}}_k = \{D_{tr}, D_{te}\}_k$ where D_{tr} is the training data corresponding to task k and D_{te} refers to test data.

Approach	Stage	Amount	Total number of images after the stage
Offline Bulk Data Generation (SSL)	Crystal Shapes (3D)	20	20
	Defect Types (3D)	3	60
	Relaxation Types (3D)	2	120
	Bragg Peaks (3D)	4	480
	Pattern Orientation (3D)	24	11,520
	Front Angle Central Slice (2D)	15	172,800
Online Sequential Data Generation (CL/NIL)	Crystal Shapes (3D)	1	1
	Defect Types (3D)	1	1
	Relaxation Types (3D)	2	2
	Bragg Peaks (3D)	2	4
	Pattern Orientation (3D)	24	96
	Front Angle Central Slice (2D)	9	864
	Iterations	10	8,640

TABLE I
AMOUNT OF THE DATA FOR EACH STAGE OF OFFLINE BULK AND ONLINE SEQUENTIAL DATA GENERATION.

At an instant k, we seek to assimilate information about defects represented by $\boldsymbol{\mathcal{Q}}_k$ into the model g (a DNN model with weights $\boldsymbol{\theta}_k$) such that the model can detect defects in the crystal images. However, as discussed earlier, we seek to reduce forgetting on tasks $[0, k-1]$. To facilitate this learning, we leverage the framework proposed in [21], where the CL optimization problem is written as a two player sequential game such that

$$\min_{\boldsymbol{\theta}_k \in \Omega_\theta} \quad \max_{\Delta \boldsymbol{x}_k^{(i)} \sim p(\boldsymbol{\mathcal{Q}})} \quad \left[H(\Delta \boldsymbol{x}_k, \boldsymbol{\theta}_k^{(i)}) \right]. \tag{1}$$

To perform this learning, our algorithm follows a two step paradigm, where we first use gradient ascent to approximate the cost function $\left[H(\Delta \boldsymbol{x}_k, \boldsymbol{\theta}_k^{(i)}) \right]$. To perform this cost value approximation, we define a experience replay array $\mathcal{E}$. At every new task instant k, we store samples from the training data into $\mathcal{E}$, a database of all tasks the model has observed. Eventually, with repeated gradient ascent steps, we may solve the inner optimization problem as $\mathcal{J} = \max_{\Delta \boldsymbol{x}_k^{(i)} \sim p(\boldsymbol{\mathcal{Q}})} \left[H(\Delta \boldsymbol{x}_k, \boldsymbol{\theta}_k^{(i)}) \right]$, obtaining

$$\min_{\boldsymbol{\theta}_k \in \Omega_\theta} \quad E_{\mathcal{E}} \left[\mathcal{J} \right]. \tag{2}$$

In the second step, with full knowledge of $\mathcal{J}$, we update the weights of the model g through gradient descent steps such that the objectives of defect detection on the crystal images can be satisfied. The work in [21] guarantees that this two-step strategy will converge to a minimum that balances the memory of detecting defects on the previously observed images while continuously learning to identify new defects.

IV. EXPERIMENTS

Our experiments were conducted on the *Swing* computing cluster of Argonne National Laboratory. We employed compute nodes belonging to the *gpu* partition, which are outfitted with 8 NVIDIA A100 GPUs and 320 GB GPU memory per node. All nodes are connected to each other by an Infiniband HDR interconnection network.

A. Model architecture

For all our experiments, we utilize a model with two convolutional and two feedforward layers. Each convolutional

layer is comprised of max pool operator, and we utilize relu activation function.

B. Model training

We develop three different realizations of training our model:

1) Single shot learning (SSL): Here, we rely on data that has been generated offline as described in Section III with no sequential data generation. Our capacity to identify defects is limited to defects available in the training data as we perform the model training after all the training data are generated.
2) Naive incremental learning (NIL): Here, we naively update the model with every new defect observed in the sequential data generation. This strategy is expected to incur forgetting and quick generalization when encountering new defects that are not present in the training data.
3) Continual learning (CL): In this approach, we adapt our model with the CL framework proposed in this paper, where we aim to achieve a balance between forgetting and generalization.

C. Metrics

We utilize two metrics for evaluating the accuracy of the approaches in identifying defects. The first metric is the task accuracy—the accuracy of detecting defects on a single task (sequential iteration)—and the second one is memory accuracy—the accuracy of detecting defects on all tasks that have been observed until now. Task and memory accuracies are identical for SSL, since the training is performed after all the data are generated.

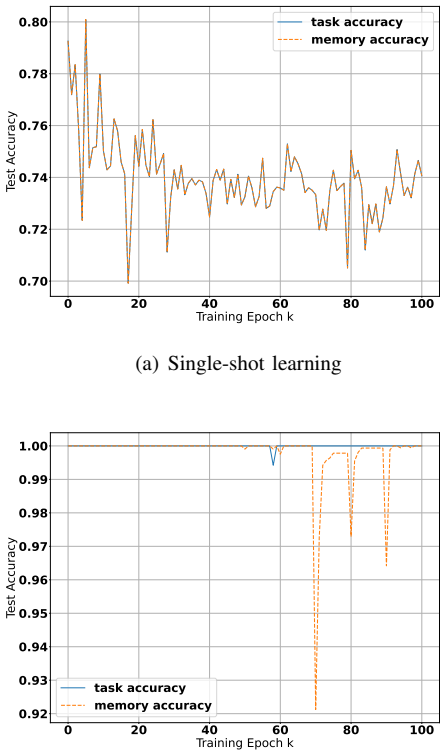
We measure task and memory accuracies on the test data set, where we hold out 15% of the total data set as test data for all approaches.

We also compare the strategies we consider in this work in terms of number of crystal images and training time required.

V. Results

First, we compare our CL approach against SSL. For the training data, Table I summarizes the data generation steps for both approaches. In CL, we generate new crystal data at every 10 epochs of training, and we repeat this for 10 iterations. With SSL, we train the model for 100 epochs after generating all the training data. Figure 2 displays the obtained test accuracy results during the training of both approaches. Here, high accuracy levels for both task and memory accuracy demonstrate that our CL approach finds a good balance between generalization and forgetting. On the other hand, we can see that SSL incurs a large generalization error.

Note that we do not fine-tune the neural network hyperparameters; rather we use the model out-of-the-box for all our experiments. One could argue that SSL can achieve similar accuracy values as the CL approach after fine-tuning the model. However, we can achieve high accuracy values with our CL approach out-of-the-box, and as we will see, with much fewer training samples.



(a) Single-shot learning

(b) Continual learning

Fig. 2. Accuracy comparison between single-shot learning and continual learning approaches.

The key advantage of our approach is that it reaches high accuracy in defect identification while using much fewer samples of data, as shown in Table II. With our CL approach, we use 95% fewer training data by generating data on demand while training the model. This savings can help scientists, who usually do not have access to large training data sets, to perform defect identification. Moreover, if data were being generated by physical light source experiments, then our CL approach would make much more efficient use of this valuable scientific instrument.

One drawback of our approach is the higher training time required compared with SSL. This is due to the fact that the CL algorithm requires more computation resources than SSL to reduce forgetting, the CL cost function presents additional terms not present in the original SSL paradigm. These additional terms require more computations to evaluate, thereby increasing both memory and computation requirements of CL. Our approach trades a 16% increase in computing time for a 95% decrease in data generation. When data generation is the largest bottleneck in the overall science pipeline, this can be a favorable trade-off.

To further explore the catastrophic forgetting paradigm, we compare our CL approach against the NIL approach, which

Approach	Completion time	Total data size
Single-shot learning	2,391 seconds	172,800 images
Continual learning	2,792 seconds	8,640 images

TABLE II
COMPLETION TIME AND OVERALL DATA SIZE FOR CONTINUAL LEARNING AND SINGLE-SHOT LEARNING APPROACHES FOR DEFECT CLASSIFICATION IN COHERENT DIFFRACTION IMAGING.

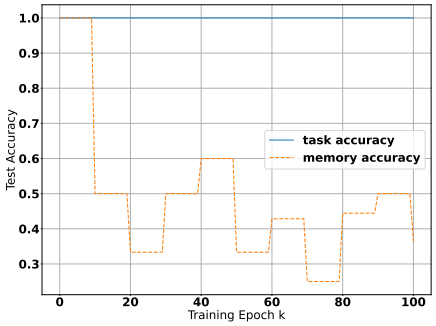
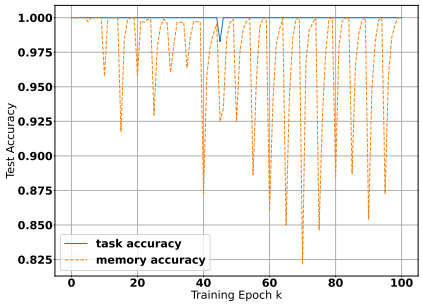


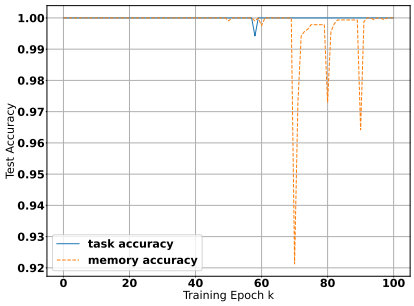
Fig. 3. Task and memory accuracy results of naive incremental learning.

performs updates to the DNN model naively when presented with sequential data. As in the previous experiment, we fed new crystal data into NIL after every 10 epochs of training, and we repeat this 10 times. Figure 3 shows the memory and task accuracy results for the NIL approach. We see that memory accuracy decreases during the run, which indicates that NIL performs poorly when it is given new crystal data. This is due to the fact that NIL generalizes quickly based on the new data as demonstrated by the high task accuracy levels in each iteration. However, NIL does not preserve the previous information and leads to catastrophic forgetting. In contrast, the results in Figure 2(b) show that we can achieve good accuracy with our CL approach, which minimizes catastrophic forgetting by virtue of explicitly modeling the impact of new task and information on the model.

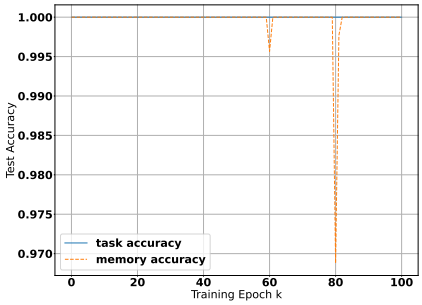
We also investigated the impact of number of epochs on the efficiency of our CL approach. Figure 4 displays the obtained test accuracies during different iterations of CL training for a total of 100 epochs, varying the number of epochs per iteration. In Figure 4(a), we generate a new crystal after training the model for 5 epochs, while we generate new data after every 10 epochs of training in Figure 4(b), and every 20 epochs in Figure 4(c). The results show that the memory accuracy dips slightly when a new task is introduced. The more frequently new data are introduced, the more the accuracy goes down. This behavior is expected, as each iteration generates data with random properties (e.g., defect type, random slicing). However, we can see that the accuracy values increase again after some number epochs of training. Based on these results, we found 10 epochs to be the optimal frequency for performing updates to the model in order to achieve good accuracy while minimizing the total number of data samples.



(a) 5 epochs



(b) 10 epochs



(c) 20 epochs

Fig. 4. Test accuracies during different iterations of CL training with different number of epochs per iteration.

VI. CONCLUSION

Defect identification is key to understanding materials' performance and properties. However, the data generation required for this process is computationally and manually intensive. In this work, we have presented an automated workflow that allows scientists to perform defect identification by using smaller data sets. To this end, we coupled data generation with training and inference of deep neural network defect classifiers. In particular, we employed a continual learning approach, where we generated training data on demand while a neural network is being trained. We evaluated our approach against single-shot learning and naive incremental learning approaches. Our results reveal that the continual learning approach learns to identify nondefective and defective crystals

to a high degree of accuracy while using much fewer samples of data.

In future work, we plan to expand our automated defect identification workflow to crystal structures with multiple defects. Another direction that we will explore involves smart data generation, where we will focus on generating crystals that are likely to improve the efficiency of automating defect identification in coherent diffraction imaging. To this end, we will explore 3D shape similarity [31].

ACKNOWLEDGMENT

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under contract number DE-AC02-06CH11357. We gratefully acknowledge the computing resources provided on Swing, a high-performance computing cluster operated by the Laboratory Computing Resource Center at Argonne National Laboratory. Work performed at the Center for Nanoscale Materials and Advanced Photon Source, both U.S. Department of Energy Office of Science User Facilities, was supported by the U.S. DOE, Office of Basic Energy Sciences, under Contract No. DE-AC02-06CH11357.

REFERENCES

[1] A. Ulvestad, M. Menickelly, and S. Wild, "Accurate, rapid identification of dislocation lines in coherent diffractive imaging via a min-max optimization formulation," *AIP Advances*, vol. 8, no. 1, p. 015114, 2018.

[2] A. Ulvestad, Y. Nashed, G. Beutier, M. Verdier, S. Hruszkewycz, and M. Dupraz, "Identifying defects with guided algorithms in bragg coherent diffractive imaging," *Scientific reports*, vol. 7, no. 1, pp. 1–9, 2017.

[3] B. Lim, E. Bellec, M. Dupraz, S. Leake, A. Resta, A. Coati, M. Sprung, E. Almog, E. Rabkin, T. Schulli *et al.*, "A convolutional neural network for defect classification in bragg coherent x-ray diffraction," *npj Computational Materials*, vol. 7, no. 1, pp. 1–8, 2021.

[4] S. Marchesini, "Invited article: A unified evaluation of iterative projection algorithms for phase retrieval," *Review of scientific instruments*, vol. 78, no. 1, p. 011301, 2007.

[5] M. J. Cherukara, Y. S. Nashed, and R. J. Harder, "Real-time coherent diffraction inversion using deep generative networks," *Scientific reports*, vol. 8, no. 1, pp. 1–8, 2018.

[6] A. Scheinker and R. Pokharel, "Adaptive 3d convolutional neural network-based reconstruction method for 3d coherent diffraction imaging," *Journal of Applied Physics*, vol. 128, no. 18, p. 184901, 2020.

[7] L. Wu, P. Juhas, S. Yoo, and I. Robinson, "Complex imaging of phase domains by deep neural networks," *IUCrJ*, vol. 8, no. 1, pp. 12–21, 2021.

[8] R. Harder, "Deep neural networks in real-time coherent diffraction imaging," *IUCrJ*, vol. 8, no. Pt 1, p. 1, 2021.

[9] H. Chan, Y. S. Nashed, S. Kandel, S. O. Hruszkewycz, S. K. Sankaranarayanan, R. J. Harder, and M. J. Cherukara, "Rapid 3d nanoscale coherent imaging via physics-aware deep learning," *Applied Physics Reviews*, vol. 8, no. 2, p. 021407, 2021.

[10] Y. Yao, H. Chan, S. Sankaranarayanan, P. Balaprakash, R. J. Harder, and M. J. Cherukara, "Autophasenn: unsupervised physics-aware deep learning of 3d nanoscale bragg coherent diffraction imaging," *npj Computational Materials*, vol. 8, no. 1, pp. 1–8, 2022.

[11] E. Deelman, D. Gannon, M. Shields, and I. Taylor, "Workflows and e-science: An overview of workflow system features and capabilities," *Future generation computer systems*, vol. 25, no. 5, pp. 528–540, 2009.

[12] I. Altintas, S. Purawat, D. Crawl, A. Singh, and K. Marcus, "Toward a methodology and framework for workflow-driven team science," *Computing in Science & Engineering*, vol. 21, no. 4, pp. 37–48, 2019.

[13] U. Ayachit, A. Bauer, B. Geveci, P. O'Leary, K. Moreland, N. Fabian, and J. Mauldin, "ParaView Catalyst: Enabling in situ data analysis and visualization," in *Proceedings of the First Workshop on In Situ Infrastructures for Enabling Extreme-Scale Analysis and Visualization*. ACM, 2015, pp. 25–29.

[14] T. Kuhlen, R. Pajarola, and K. Zhou, "Parallel in situ coupling of simulation with a fully featured visualization system," in *Proceedings of the 11th Eurographics Conference on Parallel Graphics and Visualization (EGPGV)*, 2011.

[15] D. A. Boyuka, S. Lakshminarasimham, X. Zou, Z. Gong, J. Jenkins, E. R. Schendel, N. Podhorszki, Q. Liu, S. Klasky, and N. F. Samatova, "Transparent in situ data transformations in adios," in *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*. IEEE, 2014, pp. 256–266.

[16] U. Ayachit, B. Whitlock, M. Wolf, B. Loring, B. Geveci, D. Lonie, and E. Bethel, "The SENSEI generic in situ interface," in *Proceedings of the 2nd Workshop on In Situ Infrastructures for Enabling Extreme-scale Analysis and Visualization*. IEEE Press, 2016, pp. 40–44.

[17] M. Dorier, G. Antoniu, F. Cappello, M. Snir, R. Sisneros, O. Yildiz, S. Ibrahim, T. Peterka, and L. Orf, "Damaris: Addressing performance variability in data management for post-petascale simulations," *ACM Transactions on Parallel Computing (TOPC)*, vol. 3, no. 3, p. 15, 2016.

[18] O. Yildiz, M. Dreher, and T. Peterka, "Decaf: Decoupled dataflows for in situ workflows," in *In Situ Visualization for Computational Science*. Springer, 2022, pp. 137–158.

[19] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[20] G. M. Van de Ven and A. S. Tolias, "Three scenarios for continual learning," *arXiv preprint arXiv:1904.07734*, 2019.

[21] K. Raghavan and P. Balaprakash, "Formalizing the generalization-forgetting trade-off in continual learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 17 284–17 297, 2021.

[22] R. Krishnan and P. Balaprakash, "Meta continual learning via dynamic programming," *arXiv preprint arXiv:2008.02219*, 2020.

[23] P. Hirel, "Atomsk: A tool for manipulating and converting atomic data files," *Computer Physics Communications*, vol. 197, pp. 212–219, 2015.

[24] S. Plimpton, P. Crozier, and A. Thompson, "Lammps-large-scale atomic/molecular massively parallel simulator," *Sandia National Laboratories*, vol. 18, p. 43, 2007.

[25] V. Favre-Nicolin, G. Girard, S. Leake, J. Carnis, Y. Chushkin, J. Kieffer, P. Paleo, and M.-I. Richard, "Pynx: high-performance computing toolkit for coherent x-ray imaging based on operators," *Journal of Applied Crystallography*, vol. 53, no. 5, pp. 1404–1413, 2020.

[26] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[27] P. Hirel, "Atomsk: A tool for manipulating and converting atomic data files," *Computer Physics Communications*, vol. 197, pp. 212–219, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.cpc.2015.07.012

[28] S. Plimpton, "Short-Range Molecular Dynamics," *Journal of Computational Physics*, vol. 117, no. 6, pp. 1–42, 1997. [Online]. Available: http://www.cs.sandia.gov/sjplimp/main.html

[29] V. Favre-Nicolin, J. Coraux, M.-I. Richard, and H. Renevier, "Fast computation of scattering maps of nanostructures using graphical processing units," *Journal of Applied Crystallography*, vol. 44, no. 3, pp. 635–640, 2011.

[30] D. B. Williams and C. B. Carter, "The transmission electron microscope," in *Transmission electron microscopy*. Springer, 1996, pp. 3–17.

[31] H.-Y. Shum, M. Hebert, and K. Ikeuchi, "On 3d shape similarity," in *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 1996, pp. 526–531.