Machine Learning Applications in Material Science Problems

by

Mingren Shen

A dissertation submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

(Materials Science and Engineering)

at the

University of Wisconsin- Madison

2021

Date of final oral examination: 11/15/2021

The dissertation is approved by the following members of the Final Oral Committee:

Dane Morgan, Professor, Department of Materials Science and Engineering
Izabela Szlufarska, Professor, Department of Materials Science and Engineering
Maria K. Chan, Scientist, Center for Nanoscale Materials of Argonne National Laboratory
Victor Zavala, Professor, Department of Chemical and Biological Engineering
Paul Evans, Professor, Department of Materials Science and Engineering

Abstract

Machine learning tools have the potential to provide a new solution for problems in material science community. In this thesis, I will present my works about applying machine learning methods to solve two typical material sciences problems, one is the defect detection problem and another one is the X-ray image pattern problem. Chapter 1 is an introduction of the thesis that states the goal of this thesis and key concepts learned in my Ph.D. study. Chapter 2 talks about the important background knowledge about machine learning, deep learning, and computer vision which are frequently used later. In Chapter 3, three deep learning based defect analysis systems are discussed for TEM/STEM images or videos. Those models prove the ability of deep learning models and show the potential of applying them to solve defect detection problems. In Chapter 4, we introduce a deep learning based classifier that can assist the interpretation of X-ray image patterns which paves the way to better understand the patterns. Chapter 5 summarize other published work I completed at UW-Madison which were not closely related to material science but shared the general theme of this thesis. Finally, in Chapter 6, a summary and future of work is present.

Acknowledgements

I would like to first express my sincere thanks to my advisor, Professor Dane Morgan, at University of Wisconsin - Madison. Without Prof. Morgan's continual support and guidance, this thesis could not be finished. During my Ph.D. study, there are ups and downs and Prof. Morgan always gives me the freedom to explore my interest and points key things either in research or other career-building processes. Prof. Morgan sets a great example and high standard for me not only in the Ph.D. study both also in project management, English communications, and public speech. I owe great thanks and sincere gratitude to Prof. Morgan. I also have many thanks to my co-mentors, Dr. Maria Chan, in Argonne National Laboratory. She shows me a great example of how to be excellent scientists, as she always shows me great amounts of knowledge and passion in the machine learning for material sciences field. I'd also like to thank Prof. Kevin Field for giving me the great opportunity to work in defect defection for TEM images. It is also a great honor for me to have Prof. Izabela Szlufarska, Prof. Victor Zavala, Prof. Paul Evans for being on my thesis committee and providing many valuable insights at pivotal points throughout my Ph.D. research.

There are many other collaborations, directly or indirectly related to my works, I'd like to express my gratitude and appreciation. I thank Dr. Dina Sheyfer, Mr. Wei Li, Dr. Ryan Jacob, Mr. Guanzhao Li, Mr. Dongxia Wu, Ms. Yuhan Liu, Dr. Troy David Loeffler, Dr. Sankaranarayanan Subramanian, and Dr. Brian Stephenson as direct collaborators in my works, giving me many helpful supports and advises. I would like to thank all members and alumni of Computational Materials Group, especially Dr. Yeqi Shi, Dr. Zhuohan Li, Dr. Hongliang Zhang, Dr. Ryan Jacob, Dr. Jianqi Xi, Dr. Benjamin Afflerbach, Mr. Shuguang Wei, and Mr. Lane Enrique Schultz for their help in my Ph.D. studies and research activities. I am proud and grateful that I have worked

with all these excellent people who make the Ph.D. study interesting and enjoyable. I will not forget all the great memory with you.

Most importantly, I want to thank my beloved parents, Ms. Jiahe Zhang and Mr. Tongtao Shen, who have given me so much love, support, and encouragement. It is their unconditional trust, timely encouragement, and endless patience that help me get through the Ph.D. study. This thesis is dedicated to you.

Mingren Shen

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ATAT: Alloy Theoretic Automated Toolkit

ML: Machine Learning

DL: Deep Learning

CV: Computer Vision

CNN: Convolutional Neural Networks

SHAP: SHapley Additive exPlanations

GPU: Graphics Processing Unit

TPU: Tensor Processing Unit

IoU: Intersection over Union

TEM: Transmission Electron Microscopy

STEM: Scanning Transmission Electron Microscopy

YOLO: You Only Look Once

Faster R-CNN: Faster Regional Convolutional Neural Networks

Mask R-CNN: Mask Regional Convolutional Neural Networks

CLAHE: Contrast Limited Adaptive Histogram Equalization

SGD: Stochastic Gradient Descent

Dpa: displacements per atom

IVEM: Intermediate Voltage Electron Microscope

NMS: Non-Max Suppression

IDL: Interactive Data Language

GAN: Generative Adversarial Networks

AR: Augmented Reality

CAM: Class Activation Map

Gard-CAM: Gradient-weighted Class Activation Map

ILSVRC: ImageNet Large Scale Visual Recognition Challenge

Chapter 1 : General Introduction

1.1 Overview of the Thesis

This thesis is comprised of six chapters. Chapter 1 provides a brief introduction of the work contained here, which introduces the overview and objectives of this thesis. Chapter 2 focuses on a description of the key theoretical concepts, mechanisms, and approaches of three important methods, machine learning (ML), deep learning (DL), and computer vision (CV), used in this thesis. In Chapter 3, we discussed how to apply deep learning models, Faster Regional Convolutional Neural Networks (Faster R-CNN)[1], You Only Look Once (YOLO)[2], and Mask Regional Convolutional Neural Networks (Mask R-CNN)[3], to solve the defect detection problem in electron microscopy images and videos. Chapter 4 expands the deep learning vision tools to explore the coherent X-ray imaging problems. In Chapter 5, other research achievements that are not material science oriented I completed in UW-Madison are discussed. Chapter 6 is about a summary of the whole thesis and future directions that are worth exploring.

1.2 Motivation for the current investigation

Material scientists usually rely on three categories of tools which can be roughly called theoretical methods, experimental methods, and computational methods to discover new materials and study their properties. However, since 2012, machine learning and deep learning based tools or called the data-driven methods or materials informatics are becoming more and more popular in the science community which is even called the Four Paradigms of Science[4,5]. This change is mainly due to three driving factors including, cheaper and more powerful hardware like Graphics Processing Unit (GPU) and Tensor Processing Units (TPUs)[6], bigger and easy to access data like

Materials Genome Initiative (MGI)[7,8], and more efficient and smarter algorithm like back-propagation and convolutional (CNN), and recurrent (RNN) neural networks[9,10].

1.3 Thesis Objectives

Motivated by the growing success of data driven and deep learning based methods, in this thesis, I will present how to apply three different deep learning models called Faster R-CNN, YOLO, and Mask R-CNN to detect objects in electron microscopy images or videos and build mappings between coherent X-ray imaging patterns with sample information like number density. The goal of this thesis is to prove that we can benefit from data driven methods like deep learning and machine learning to study material science problems.

1.4 Key Lessons learned when applying ML to material sciences

Before we discuss details about machine learning and how to apply it to material sciences, I want to review three important lessons I learned during my Ph.D. study. It is critical to build basic principles about how to apply machine learning and deep learning since we want to benefit from the advantages of ML and avoid the shortcomings.

Applying ML methods belongs to the study of computer sciences so common wide-used practices of computer science and software engineering e.g., unit testing, version control, and interface design are also useful, but it is not the focus of this thesis and we recommend readers to read pieces of literature listed for more information[11–15]. Besides knowledge from software engineering, some special lessons related with applying ML to material sciences problems are worth discussions. Below I listed three important tips that help me most and I will discuss them one by one.

- **No Free Lunch Theorem**

  Since the growing success of ML methods, researchers tend to apply ML to new problems without any hesitations, however, in the ML community, there is a famous lesson called No Free Lunch Theorem which states that there is no best algorithm for all problems in optimization and due to the close relationship between optimization and ML, no best algorithm works for all ML problems[16]. Luckily, we don't need to build an ML algorithm that works best for all problems, with certain assumptions and constraints of problem spaces, we shall be able to pick the algorithm that works best in current constraints. When applying ML into material sciences, we shall keep in mind and reflect on the problem design repeatedly, what is the final goal of the study, what is the type of problem we are studying, and finally, what is the assessments metrics in our study. For example, if the final problem needs thorough explanations, then decision trees might be better than complex neural networks although more and more explanation tools are developed for neural networks such as SHapley Additive exPlanations (SHAP) values[17,18], the decision tree algorithm is still the most efficient tool to explain the working mechanism of ML[19,20].

- **Domain Knowledge Matters**

  Due to the rapid development of ML tools, once domain-related data is obtained, it is usually easy to launch a ML study and generate some results. However, how to understand these results? Are they useful? What is the insight or guidance we can provide? For example, based on your ML model you can well predict a certain property of material but in the material and science community, people want you to better explain why this model

work and how to understand the working mechanisms from the physical principal level. An extended example might be physics around 1900. If you follow the common practice in ML, it would be easy for you to conclude that there are no more important tasks to do which, as Lord Kevin said, “nothing new in physics and only more precise measurements might be needed and only two dark clouds still shown in the sky of physics”[21]. And we all know the following story, these two dark clouds turned out to yield special relativity and quantum mechanics[21]. There are two different methodologies here, exploration, finding new things, and exploitation, fitting already known. And a common trend in the ML community is always trying to fit things as precise are possible and sometimes outliers are easily ignored. However, in the material sciences community, sometimes those outliers are the key which might be useless in one metric but valuable in another metric.

- **Data Matters**

  Algorithm, Hardware Improvements, and Dataset Growth are well recognized as the driving forces of the success of deep learning recently[9]. However, if your time or resources are limited, I highly recommend growing the dataset as the first option of solutions, especially for material science problems. ML and DL are essentially learning things from the dataset so if the dataset is problematic or small, the power of algorithms is also limited or biased. As the famous saying goes, “Garbage In, Garbage Out”, so do the ML models. And if time and resources permit, using simulation or other computational methods to generate a clean labeled dataset will be the most important key to building a successful ML project in material science and related projects.

Chapter 2 : Backgrounds

2.1 Chapter Abstract

In this chapter, I will present a short introduction to three aspects of key background knowledge that are important for this thesis. In **Section 2.2**, a short introduction of machine learning is provided, in **Section 2.3**, one special type of machine learning tasks called deep learning is provided, and finally, in **Section 2.4**, a specific problem called computer vision is discussed and currently the majority of methods in computer vision is deep learning based.

2.2 Machine learning

According to Mitchell, machine learning is defined as, for certain task ***T*** and performance measure ***P***, an algorithm that can learn from past experience ***E*** to increase its performance measure ***P*** for task ***T***[22]. For example, a classification task ***T*** could be classifying images of dogs and cats, and performance measure ***P*** would be the accuracy that defined as correct classified testing images over the total number of testing images, and then the experience ***E*** would be the training process of feeding a group of already labeled images of cats and dogs into the algorithms saying neural network that could improve its performance measure ***P*** using back-propagation methods[22]. Machine learning has a lot of details that are clearly out of the scope of a thesis, so I refer readers to the nice introduction paper by Prof. Jordan and Prof. Mitchell as a starting point[23].

There are multiple ways to categorize machine learning methods and one of the widely used categorizations is dividing machine learning methods into 3 categories based: supervised learning, unsupervised learning, and reinforcement learning[23]. And in this thesis, I mainly use supervised learning so I will briefly discuss it here. In supervised learning, for each instance of training sample X_i, it will have a predefined label or value assigned to it called y_i. the learning

process is to learn a proper mapping between X and y. For example, for the classifier for images of cats and dogs, each image will have the label to tell the ML algorithm whether the image contains dogs or cats, and the learning process can improve its weights based on the error between true label and prediction.

2.3 Deep Learning

Deep learning typically uses a combination of multiple layers of nodes called neural networks to extract the intrinsic structure of input data to build a mapping between the intrinsic structure and targeted output[9]. With advancements in GPU computing powers, accumulation of carefully labeled large scale datasets, e.g. ImageNet[24], and better optimization algorithms like backpropagation, deep learning-based models have shown great success in different tasks such as automatic driving, speech synthesis, and image classifications[9], even outperforming humans in many tasks such as the board game Go[25]. Deep learning has also been widely used in material science[26] and achieves good performances including predicting properties of materials[27,28], identifying material phase transitions[29], and automating the analysis of TEM/STEM data[30–36].

2.4 Computer Vision

One of the most successful applications of deep learning is computer vision, where the ultimate goal is teaching a computer to do the image-related task(s) like finding which object is contained in an image (object detection) and which pixel belongs to different objects (image segmentation)[37]. Since a breakthrough in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) was made in 2012, the deep Convolutional Neural Network (CNN) based

approach has demonstrated its success in many image-related tasks[37]. There are nice books[38,39] about the detailed techniques and progress of computer vision. In this thesis, we focus on one specific problem of computer vision, object detection and tracking, which means teaching the computer to know the types and positions of objects contained in images or videos. And we refer the readers to those good reviews for more details[37,40,41].

For the object detection problem (trying to find the location and category of all the objects contained in an image), which is also the focus of **Chapter 3**, there are two general categories of methods: two-stage methods and one-stage methods[37]. For a two-stage method like Faster R-CNN[42], the object detector will first propose some candidate bounding boxes containing the object location information and then classify the category of those candidate bounding boxes. One-stage methods like YOLO will output the object location and category at the same time[43]. Typically, two-stage methods are slower but more accurate than one-stage methods. YOLO is one of the most widely used one-stage methods and offers speed, accuracy, and fast engineering application potentials[37]. The key ideas of YOLO are dividing the whole image (or video frames) into grids and predicting the location and the category of the potential bounding boxes with a set of pre-defined anchor boxes in each cell of the grid[43]. Other widely used models for object detection problems are U-Net Family and Transformer Family methods. In U-Net, the network has a U shape structure where the left branch would encode the key information into a smaller inner vector space and the right branch would decoder the inner representation to generate pixel level segmentation maps[44–48]. U-Net is easy to implement and good at working with small datasets and thus is popular in medical image applications[49]. Transformer based methods using positional encoding to split each image into patches and align these patches into ordered sequences to apply natural language processing methods for the object detection problem and

Transformer family methods are actively studied in computer vision community now since it could easily encode global information and correlations into object detection mechanism[50–52].

Chapter 3 : Defect Detection

3.1 Chapter Abstract

Defect detection in material is counting the number of different types of defects and collecting their geometric information such as total area, mean radius from electron microscopy images. Defect detection is important since it can reflect the property changes in materials. For example, the simplified dispersed barrier hardening model can help calculate the hardening due to defects[53]. In practice, defect detection is mainly done by humans to calculate the number and distribution of each defect type in each image and then summarize the corresponding hardening due to defects. But this process is tedious, error-prone, bias-involved, and non-reproduced. In this chapter, I will present three deep learning based defect analysis systems, Faster R-CNN, YOLO, and Mask R-CNN I developed to help alleviate the difficulty of defect detection.

3.2 Defect Analysis of STEM/TEM Images

3.2.1 Introduction

Analyzing the locations and sizes of defects in materials that have undergone irradiation is a widely used application of electron microscopy. In such studies, the key properties are the total number and distribution of each type of defect. Typical defects of interest include grain boundaries, precipitates, dislocation lines, dislocation loops, stacking fault tetrahedra, cavities (voids, bubbles), and co-called “black-spot” defects, which are small defect clusters of interstitials and sometimes vacancies[54,55]. For this study, we focus on the dislocation loops formed within a ferritic alloy, where the loops exist on specific habit planes that manifest themselves with different morphologies due to the projection of a 3D volume imaged using

EM[56]. Typical microstructural images of irradiated ferritic steels contain four prominent types of defects: (1) open ellipse loops (single ring edge), (2) open ellipse loops (double ring edges), (3) closed solid elliptical loops, (4) closed circular solid dots[56]. **Figure 3.2.1** shows a sample STEM image containing all four morphologies of loops obtained from a ferritic alloy irradiated in a materials test reactor.

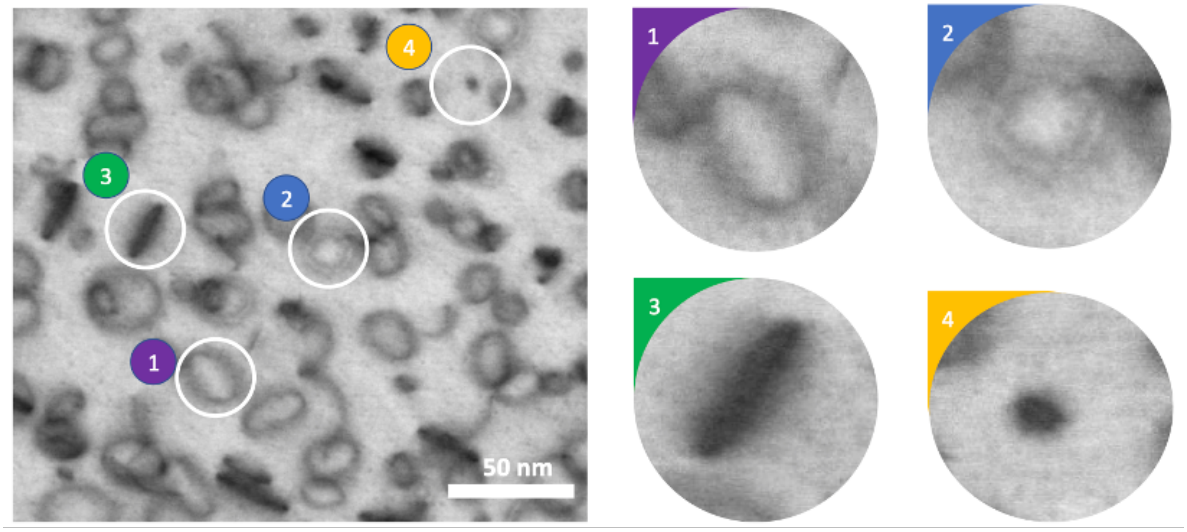

Figure 3.2.1. Selected bright field scanning transmission electron microscopy (STEM) image of an irradiated ferritic alloy showing four common morphologies of dislocation loops: (1) open ellipse loops (single ring edge), (2) open ellipse loops (double ring edges), (3) closed elliptical solid loops, (4) closed circular solid dots. Open single edge ellipse loops (1) are dislocation loops with a Burgers vector of $^{a_0}/_2\langle 111\rangle$. Open double edge ellipse loops (2) and closed elliptical solid loops (3) are dislocation loops with a Burgers vector of $a_0\langle 100\rangle$. Closed circular solid dots (4) are black dot defects with a Burgers vector of either $^{a_0}/_2\langle 111\rangle$ or $a_0\langle 100\rangle$. Image size: Primary image is 290 × 290 nm; inset scales arbitrary.

3.2.2 Methods

In this section, we used a modern deep learning-based object detection model called Faster Regional CNN (Faster R-CNN)[42], a widely used deep learning based object detection model[57]. We use the Faster R-CNN to develop an automatic defect detection system for all four morphologies commonly observed in irradiated steels with a body-centered cubic structure and then additional post-processing to analyze their geometrical information (specifically, size and areal density). This paper serves to demonstrate the power of deep learning-based computer vision models for material image studies and suggests the possibility that most aspects of defect analysis may soon be practically automated, and many, if not all, handcrafted feature-based methods may be replaced by deep learning methods.

Faster R-CNN is a CNN based end-to-end deep learning object detection model that outputs both the object position and its class[42]. As shown in **Figure 3.2.2**, Faster R-CNN is a two-stage detector where the region proposal network (RPN) proposes Region of Interest (ROI), and the following ROI regressor and classifier will fine tune the final output results including the size and position of the object contained bounding boxes and the corresponding object label[42]. Given an image, the shared convolutional layers will extract a feature map from the input image by performing a series of convolution and max pooling operations. Then based on the extracted feature map, the RPN will put a set of predefined anchor boxes on the feature map and output the probability of whether the anchor box belongs to an object of interest or plain background. It worth mentioning that RPN ignores the specific object class of each bounding box and the following ROI regressor and classifier are responsible for the specific class and refined location of the objects. The refining network predicts certain object labels and refines the size and position of each bounding box based on the feature map generated by the ROI-pooling layers[40].

The RPN and ROI components are trained jointly to minimize the loss function sums from both of them[42]. After the Faster R-CNN module A, those images with detected defects are sent to module B to extract geometric information such as defect diameters, as shown in **Figure 3.2.2**.

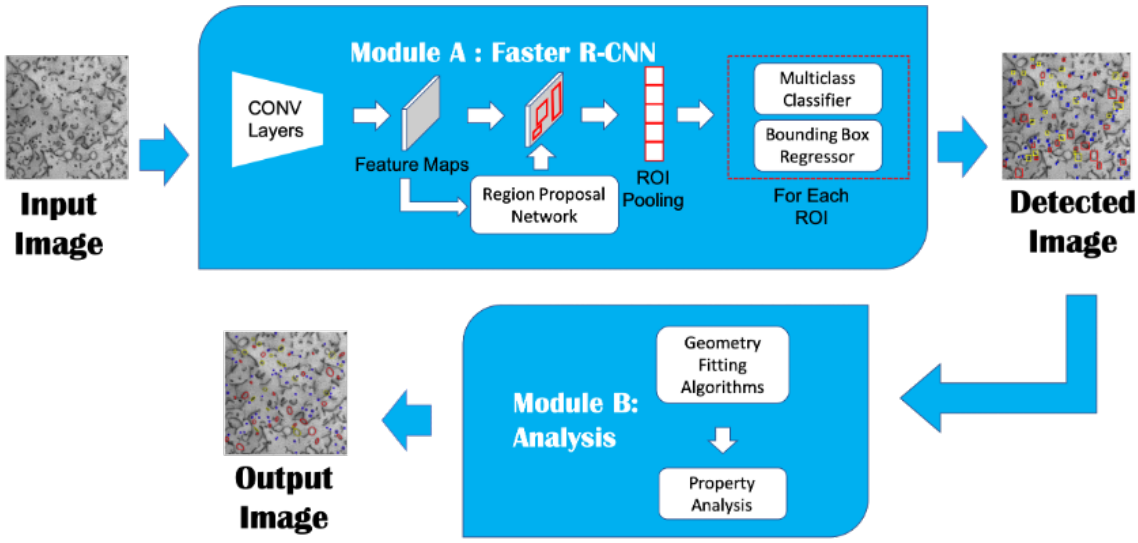

Figure 3.2.2. Schematic flow chart of proposed deep learning based automated detection approach. Input micrograph images go through the pipeline of Module A—Faster R-CNN Detector, Module B—Image Property Analysis. After Module A, the loop locations and bounding boxes are identified and then for each identified bounding box, geometry fitting algorithms are called to determine the defect shape and size in Module B.

3.2.2.1 Data Set Collection

Data set collection was completed as part of a large-scale effort to characterize iron-chromium-aluminum (FeCrAl) materials neutron-irradiated within the High Flux Isotope Reactor at Oak Ridge National Laboratory. The dataset comprises a series of published[53,58,59] and unpublished data. The data collection was completed over 3 years and spanned a range of

different FeCrAl alloys, including model, commercial, and engineering-grade alloys irradiated to light water reactor–relevant conditions (e.g., <15 displacements per atom and temperatures of nominally 285–320°C). Images generation are described in more details in Li et al[32].

3.2.2.2 Data Set Preparation

We used ImageJ[60,61], an open-source software for analysis of scientific images, to manually label all the training and testing data set. And since STEM images are gray scale and ChainerCV[62] expects input images with RGB channels, some modifications are necessary. We use the direct STEM image gray scale for the R channel. Then we use modifications of the original image gray scale for the G and B channels. Specifically, following Li et al.[32], for G we use a local contrast enhancement of the original gray scale channel saturated to maximum/minimum and for B we use a Gaussian bluer filter of the original gray scale STEM images. For the local contrast enhancement in channel G, we use the Contrast Limited Adaptive Histogram Equalization (CLAHE), a common algorithm used for local contrast enhancement that makes local detail of STEM image enhanced even in regions that are darker or lighter than most of the image. The Gaussian filter used in channel B represents cases where there might be noises or blurring in the STEM images. The parameters used for CLAHE[63] and Gaussian blur[64] are all from the default parameter setting of scikit-images and details can be found in the references given here for these methods. The purpose of adding two more channels in this way is to improve the model performance and make the model more robust by providing more information about various contrast levels or blurring.

For the training and testing on the Faster R-CNN model, a total of 165 STEM images of irradiated ferritic alloys were collected and labeled. The images were taken at different

experimental conditions of temperature and irradiation damage level so that the data includes varying defect sizes, shapes, and areal density. We constructed the ground truth labeling by giving each image in the dataset to at least two groups of at least two researchers per group who together labeled each image in that dataset. In some cases, no absolute consensus could be reached on whether a feature was a defect and/or what type it had, in which case a best effort was made based on group discussion.

The test dataset was randomly selected from the complete image dataset, so that the training and test were split by approximately 10:1 ratio. The training dataset was then augmented to 918 images in total, which could provide more training instances without spending more manpower on labeling. The data is augmented by rotating and/or flipping each image in the training set, a standard method previously well established to improved results in some cases[65].

3.2.2.3 Model Training

The Faster R-CNN model used VGG-16 as its backbone architecture and we adopted the module provided by ChainerCV[62] as the Module A in **Figure 3.2.2** and using watershed function provided by OpenCV[66] as the second module. The initial weights of Faster R-CNN was loaded from the pre-trained weights from ImageNet which is a common practice in the deep learning training strategy[67] called transfer learning. Although ImageNet is trained for image classification, not object detection, there are enough similarities in key features to support effective transfer learning of weights. Transfer learning can reduce the amount of data and training time required for good performance[67]. The Faster R-CNN module was optimized with Stochastic Gradient Descent (SGD) on a single Nvidia GeForce GTX 1080 GPU. The best hyper parameter set was found by performing hyperparameter search of learning rate from 10^{-3} to 10^{-6}

and we adjust the needed iteration numbers correspondingly. The best choice of hyper parameter is a decayed learning rate starting from 10^{-4} and each 20000 iterations the learning rate will decay to one tenth of the previous one. In total 90000 iterations were performed, and a learning loss curve is shown in **Figure 3.2.3**. The geometry extraction module needed no training.

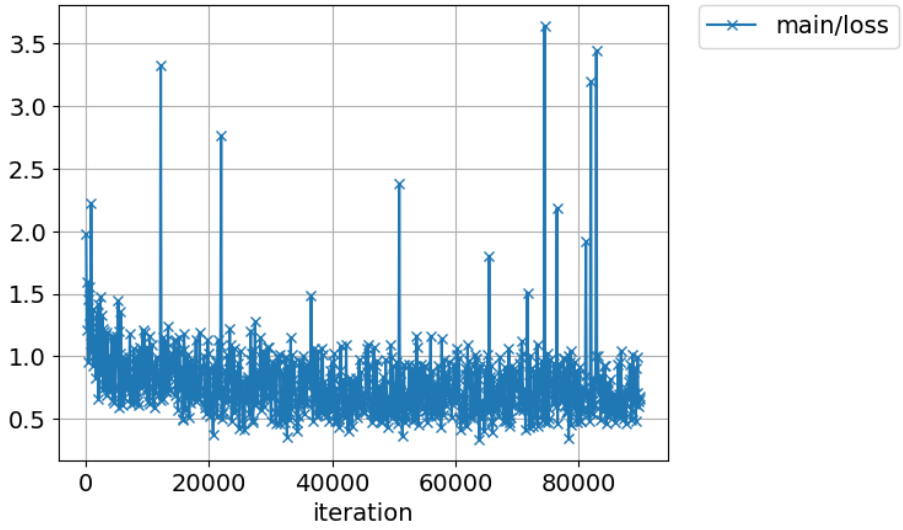

Figure 3.2.3. A typical loss curve for Faster R-CNN training.

3.2.2.4 Model Testing

After the Faster R-CNN module was trained, there were still two important hyperparameter associated with accuracy analysis: the threshold IoU value and the confidence score. IoU stands for Intersection over Union and is an evaluation metric used to measure the performance of object detection models[57]. IoU is calculated from the ratio of overlap area of a ground truth bounding box and a predicted bounding box to the area of union of two bounding boxes. The range of IoU is from 0 to 1 where 0 means no overlap found between two bounding boxes and 1 means the two bounding boxes are perfectly overlapping. The threshold IoU is the

value used to judge the prediction quality of the overlapping of ground truth bounding boxes and prediction bounding boxes. A higher threshold IoU requires more accurate location prediction of the bounding box detector, which will generally reduce performance, but lower the threshold IoU could lead a predicted bounding box to being assigned to no defect or the wrong defect. And another important hyperparameter is the threshold confidence score, a value from 0.0 to 1.0 used by Faster R-CNN internally to discard low confidence proposals in the RPN, and it can change the total number of outputs of Faster R-CNN. We used grid search of the threshold IoU and confidence score to search the best choice of these two values based on maximizing the F1 scores, with confidence score from the list [0.001, 0.005, 0.01, 0.05, 0.1, 0.15, 0.2, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.55, 0.6] and the threshold IoU from the list [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9]. We selected 0.25 as the confidence score for Faster R-CNN and showed the performance changes with 0.4 threshold IoU in **Figure 3.2.5**.

3.2.2.5 Geometry Fitting of Analysis Module

After the Faster R-CNN module was performed on specific image, the analysis module was called to obtain shape and size of the defect contained in bounding box. As shown in the third column in **Figure 3.2.4**, the approach fits the defect with elliptical contours to estimate their actual shapes and diameters. The approach uses the watershed algorithm to identify the pixels that make up the defect contour and then fit those to an ellipse. The watershed algorithm is a widely used technique for image segmentation purposes that views any gray scale image as a topographic surface where the high (e.g., white) pixel values represent peaks while the low (e.g. black) pixel values denotes valleys. The algorithm tries to grow the region areas by flooding the valleys and where different regions meet with each other are the watershed lines needed for

image segmentation[68]. Watershed methods were applied to find the boundary between defect pixels and background pixels. We followed the official tutorial from OpenCV for performing the watershed and details of the approach can be found there[69]. We then fit the boundaries found from the Watershed algorithm to an ellipse. This fitting was done to match the approach used by the radiation defect analysis community, obtain a well-defined shape with simple geometric descriptors, and smooth out the otherwise rather rough boundaries found by the Watershed algorithm. The fitting was done with OpenCV's `fitEllipse()` function[70]. All codes were based with OpenCV[66] and by applying the second module we could get precise information about the defects' position, size, and orientations. The diameters and areas of defects are defined as follows, where *a* and *b* are half the lengths of major and minor axes of the ellipse. The diameter of the a/2<111> and a<100> defects are defined as 2*a*. The diameter of the black dot is defined as twice the square root of (*ab*). The area of all defects is defined as πab. The areal density is the sum of defect areas in a set of images divided by the total area of the set of images.

3.2.3 Results

To assess the machine predictions, four types of approaches were taken. The first approach was a qualitative comparison of machine to human labeled images, where we looked for large fractions of errors, e.g., more than 40%, and for trends in errors that might indicate a major issue but made no attempt to quantify agreement. This assessment tests all aspects of the model as it compares to the ground truth human results, which include the bounding box predictions (the defect detection part of Module A in **Figure 3.2.2**), the defect type identifications (the categorization part of Module A in **Figure 3.2.2**), and the geometric shape determination (Module B in **Figure 3.2.2**). The second assessment approach was a quantitative

assessment of the ability to identify a defect, regardless of defect types. This assessment tested the defect detection part of Module A (see **Figure 3.2.2**). This assessment was a binary categorization problem and success was quantified with precision, recall, and F1 score. The third assessment was a quantitative assessment of the ability to identify a defect type once a defect had been correctly identified and tested the categorization part of Module A (see **Figure 3.2.2**). This assessment was a three-category categorization problem and was quantified using the confusion matrix with precision, recall and F1 calculated for each class. Finally, the fourth assessment was a quantitative assessment of the ability to quantify the geometric properties of defects. This assessment tested the geometric analysis of Module B (see **Figure 3.2.2**) and compared machine and human predictions of average and standard deviations in size and areal density for each defect type. We discuss each of the four assessments below and label them assessment 1-4 for clarity. In all cases the comparisons are made on the test data set described in Methods Section.

Assessment 1. After feeding the images into the Faster R-CNN detectors, the resulting detections were plotted on the original images. As shown in **Figure 3.2.4**, the red circles represent the dislocation loops with a Burgers vector of $^{a_0}/_{2}\langle 111\rangle$ (Type 1 in **Figure 3.2.1**), while the yellow and blue circles represent $a_0\langle 100\rangle$ direction loops (Type 2 and 3 in **Figure 3.2.1**) and "black dot" defects (Type 4 in **Figure 3.2.1**) respectively. The data from both human-labeled and machine detected results are plotted in the same manner. To a human observer the machine results show strong correlation of bounding box location, defect type identification, and defect shape with the ground truth human labeling which indicates the effectiveness of the proposed automatic defect detection system.

Assessment 2. The performance of the detection part of Module A (see **Figure 3.2.2**) of the trained model was evaluated in terms of precision, recall, and F1 score by comparing the

detected result with the human labeled result of the 12-image testing set, as shown in **Figure 3.2.5**. The precision describes the percentage of all machine-predicted bounding boxes that are judged to have correct positions, and the recall value describes the percentage of all human labeled defects that are identified as in a bounding box by the machine algorithm. F1 is the harmonic mean of the precision and recall which can be used to assess the overall performance of the defect location task[40]. The IoU (Intersection over Union) method was used to determine if a given defect was identified by a bounding box and is described within the provided Methods section. The cutoff IoU, which must be exceeded to consider the bounding box to have identified the defect, is a hyperparameter that can be fine-tuned based on the purpose of the object detection task[40]. **Figure 3.2.5** showed a drop in performance as the cutoff IoU increased. This trend agreed with expectations as the higher cutoff IoU meant it was harder for the predicted bounding box to be judged successful. However, setting the cutoff IoU to an extremely small threshold could lead to the problem that the predicted bounding boxes are associated with defects for which only a small part of the defect is in the bounding box, which will likely cause problems in the defect identification (Module B) of our model. As a compromise, for all the further assessments in this paper, we used cutoff IoU = 0.4 to determine when the machine predictions were considered to match a given defect. This choice kept nearly optimal performance of the detector (based on **Figure 3.2.5**) and an adequately demanding standard for predictions.

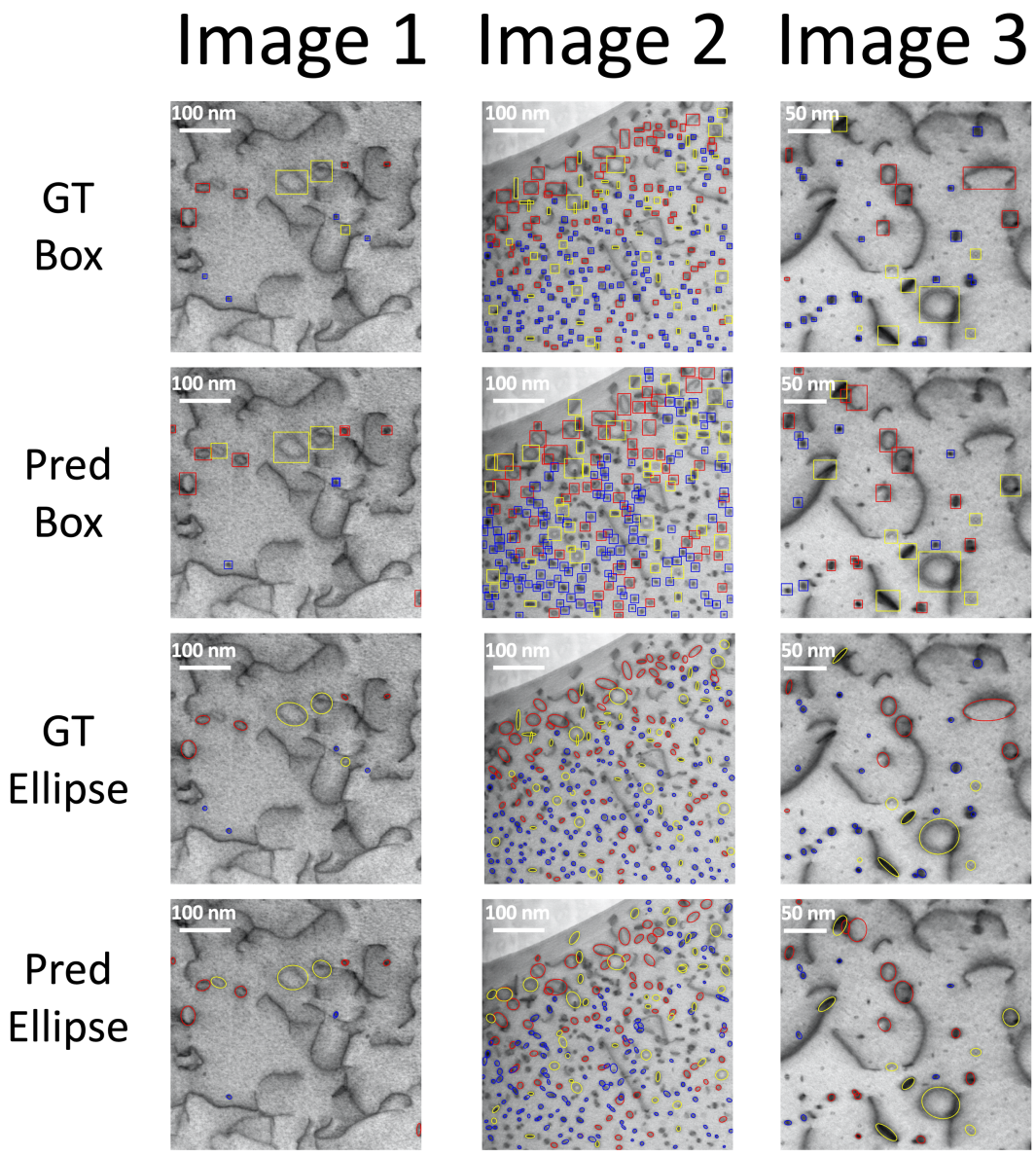

Figure 3.2.4. Selected data images to show the detector performance and the fitted ellipse of our automatic analysis system. These three test images are selected from the test dataset of 12 images (see Methods). The "Ground Truth (GT)" shows the bounding box and ellipse human labeling (colored by defect type), the "Prediction (Pred) Box" shows the predicted bounding boxes (colored by defect type), and the "Prediction (Pred) Ellipse" shows the resulting fits to the specific defect geometry (colored by defect type as described in the text).

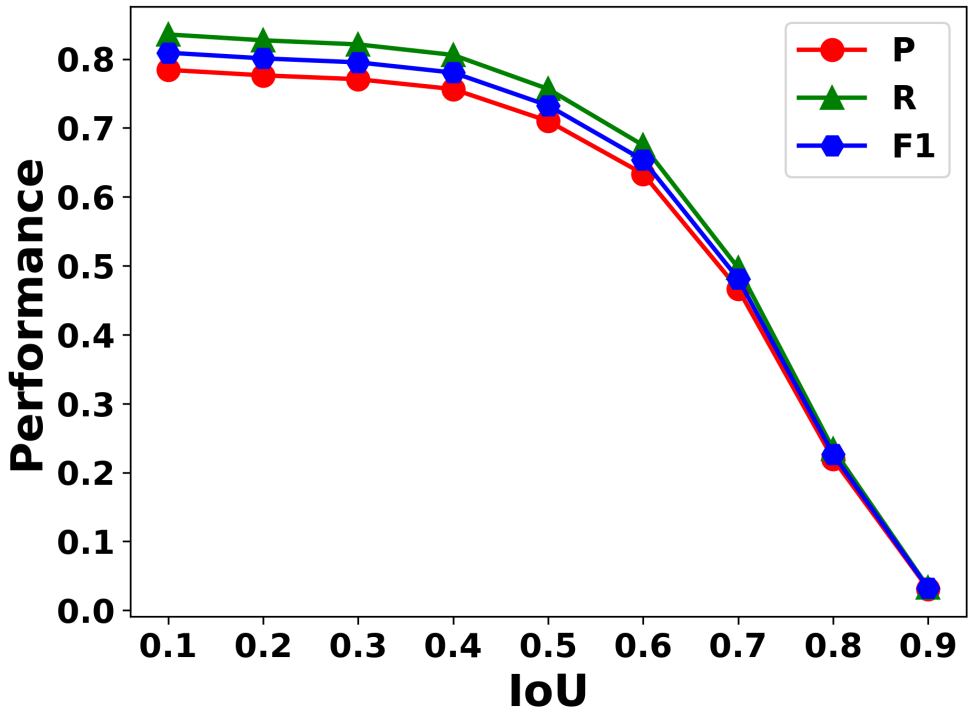

Figure 3.2.5. Summary of defect location recognition performance of all types of defects evaluated using precision and recall metrics, regardless of defect types. The test set contained 12 images, and, for all IoU (Intersection over Union) values and we used a threshold confidence score 0.25 for Faster R-CNN output. (See Method Section).

Assessment 3. **Table 3.2.1** shows the confusion matrix of the predictions made by Faster R-CNN detector evaluating its capability to correctly categorize defects. Each row in the confusion matrix represents a class that is predicted by the detector, and each column represents a class labeled by human researchers. The diagonal elements of the table represent the correct classification made by the detector and off diagonal elements represent errors of different types. We also show the percentage accuracy of each type of defect in parentheses. The 76%, 87%, and 94% accuracy indicates that once the Faster R-CNN model locates the defect, it can classify the type of defect based on their morphology within the image with good accuracy, although some

improvement of the 76% value is likely possible for the $a_0\langle 100\rangle$ loops. We also report the classification performance using precision, recall and F1 score in **Table 3.2.2**. Given inherent errors of human performance we take scores for precision, recall and F1 of 0.78 as approximately the upper limit that can be obtained with the present labeling. **Table 3.2.2** shows F1 from about 0.65 to 0.78, which demonstrates significant capabilities but is likely less than can be achieved, suggesting opportunities for further improvements.

Table 3.2.1. Summary of the classification performance for each type of defects at cutoff IoU 0.4. Values in parenthesis give the % each number represents of the total number of defects in that class as determined by the human labeling.

	$a_0/2\langle 111\rangle$ Loop	**Black Dot**	**$a_0\langle 100\rangle$ Loop**
$a/2\langle 111\rangle$ Loop	239 (87.2%)	21	14
Black Dot	17	416 (94.3%)	8
$a\langle 100\rangle$ Loop	33	13	166 (78.3%)

Table 3.2.2. The performance report for each class.

	$a_0/2\langle 111\rangle$ Loop	**Black Dot**	**$a_0\langle 100\rangle$ Loop**
Precision	0.73	0.65	0.62
Recall	0.83	0.71	0.72
F1	0.78	0.68	0.67

Assessment 4. The second Module provides geometric in formation for each defect through fitting ellipses. While the fits can provide a range of detailed information, we are particularly interested in the arithmetic mean and the associated standard deviation of the defect diameter as well as the areal density in an image for each type of defect. These values are commonly quoted values in literature within irradiated materials studies. **Table 3.2.3** compares the human labeled arithmetic mean diameters and areal densities to the ones predicted by the automatic analysis system. The discrepancy of arithmetic mean diameter between the human labeled ground truth and predictions is within 10% in all cases, which is considerably less than might be expected for variation among different humans[71] and we consider a strong success. Furthermore, the errors in arithmetic mean diameters are in the range 0.7-1.1 nm, which corresponds to a range of two to nine pixels (based on the range 0.14nm/pixel to 0.87nm/pixel for our test data). The errors of about 1 nm correspond to about 5-10% for our data which is somewhat larger than might be expected from direct labeling errors on 10-15nm. Thus, it is unlikely that any human labeling is meaningfully accurate to much below this level. However, the human and machine learning black dot radii do not fall within a 95% confidence interval, suggesting that the algorithm does not yield the same means as the human ground truth. Some errors will come from the machine detection (failures in precision and recall, see **Figure 3.2.5**) and defect type assignment (see **Table 3.2.1**). Additional errors are associated with intrinsic errors in the machine and human ellipse labeling, where both have some uncertainty due to ambiguity or variances in the morphology of defects in images. Some defects are not well fit by an ellipse (e.g., some have a more rectangular shape, as can be seen in **Figure 3.2.4**), making this form of labeling difficult for both human and machine. Another error to consider is that as the number of pixels per feature goes down, the intrinsic error due to the resolution (pixel/nm) will

artificially go up. For instance, a 100 nm loop where the resolution is 1 pixel/nm where the labeling is off by 1 pixel will yield a 1% error. If the labeling is off by 1 pixel for a 5 nm loop, the error will be 20% even though the per pixel error is the same. Seeing as the black dots are all small arithmetic mean diameter (<10 nm), they will intrinsically have a higher error compared to the other classes where the diameters are 2-3 times larger.

Table 3.2.3. Comparison of arithmetic mean defect diameter and standard deviation of mean loop diameter between ground truth labeling and our automatic analysis model prediction with an IoU of 0.4. The values in parenthesis are the relative percentage error between ground truth human labelling results and the automatic analysis results.

Defect Type	Ground Truth			Automatic Analysis Model		
	Arithmetic Mean diameter (nm)	Standard Deviation of Mean Diameter (nm)	Areal density (m^{-2})	Arithmetic Mean diameter (nm)	Standard Deviation of Mean Diameter (nm)	Areal Density (m^{-2})
$a_0/2\langle 111\rangle$ Loop	22.4	0.7	1.77×10^{14}	23.1 (3.1%)	0.8	2.21×10^{14} (24.9%)
Black Dot	8.2	0.1	3.41×10^{14}	9.1 (10.9%)	0.2	4.98×10^{14} (46.0%)
$a_0\langle 100\rangle$ Loop	20.3	0.8	1.32×10^{14}	22.4 (10.3%)	0.9	1.79×10^{14} (35.6%)

3.2.4 Discussion

The above results demonstrate that that the trained model potentially performs well enough to replace human in a workflow on similar types of data. The precision and recall values for assessing detection in the range 62-83% which are comparable or less than human variation[71] from previous assessments. The machine defect type misidentifications are at the level of 10-25% (see **Table 3.2.1**), and a significant fraction of this variation may also be due to ground truth ambiguities or errors. The final machine predicted diameters are within a nanometer, approximately 2 pixels in images, which is a level of error that is considered negligible in terms of impact on material properties. To further clarify that the error is negligible for our defect population we have done a sensitivity analysis based on previous studies of hardening from loops. As discussed in Field et al. [53] simple dispersed barrier hardening models suggest that the hardening under irradiation from loops is of the form $\Delta\sigma_y = A\sqrt{d}$ where A is a constant and d is the diameter of the defect. Now consider an error in diameter d defined as ε. The fractional error in $\Delta\sigma_y$ due to the error ε is $\left(\Delta\sigma_y(d+\varepsilon) - \Delta\sigma_y(d)\right)/\Delta\sigma_y(d) \approx \varepsilon/(2d)$, where the approximate equality holds for $\epsilon \ll d$. For ε = 1.7 nm (which is 2 pixels for our largest pixel sizes, see below) and d = 21.4 nm (our average sizes of a/2<111> and a<100> defects), we get the fractional error in $\Delta\sigma_y$ as 1.7 nm / (2 * 21.4 nm) ≈ 0.04, which is well within the uncertainty of such microstructure-based analysis. However, for smaller defects this percentage error could clearly become larger. The errors of diameter between ML results and human results appear to be approximately symmetrically distributed in positive and negative directions and independent of defect density. Furthermore, previous studies indicate that the differences of arithmetic mean diameter between different human labelers can be comparable or larger than values found here between the ML and human results[32]. The discrepancy in areal densities is somewhat larger than

might be intuitively expected just from the percentage error in the arithmetic mean diameters. However, additional errors are introduced by the exact definition of areal density (see Methods section) and the additional errors introduced by the imperfect precision and recall.

While the exact performance of the present automated approach compared to different human researchers is difficult to determine rigorously there is no doubt that the present approach is much more consistent. Previous studies have shown that different labelers tend to label defects in different ways and even the same person may label the same defects differently even after a short break[32,71]. Such issues can make any given data analysis somewhat unreliable and make it difficult to integrate results across different teams and or time periods in larger analysis efforts. However, once a machine learning model is properly trained, it will yield a unique and reproducible labeling for every image. If the community could converge on a single or small number of models this could greatly increase the reproducibility in labeling of STEM experiments. That said, models trained on different data and/or different human labeling could give different predictions, so establishing community accepted models is an important part of using these approaches to obtain more consistent results.

The approach applied here is readily scalable to very large data sets. Analyzing a single image with our model on a reasonable state of the art GPU (NVIDIA's GeForce GTX 1080 GPU) takes about 0.1s, so analyzing all the images in a typical experiment can be done easily in minutes, even less if multiple GPUs are used and as GPU and related processors (e.g., TPU) continue to get faster. As large scale distributed cloud service provider like Google, Amazon and Microsoft are providing cloud service for deep learning applications with GPU machines[72], it would be easy to scale to process even larger amount of data. Furthermore, significant speedup can likely be obtained if desired. We developed the system with the Python code language and

the ChainerCV deep learning framework, both of which were chosen for ease of development not for the high-performance in deployment. Replacing Python with C/C++ or using high-performance deep learning frameworks, e.g. Caffe[73], could potentially accelerate the prediction speed of the current model. In particular, the deep learning community is actively designing new methods to accelerate the running speed of model e.g. model compression, weight sharing, or parameter pruning[74] which could also boost the speed of ours. As an example of how fast deep learning AI algorithms can be, researchers from Google have recently applied deep learning models for cancer diagnosis on data during the actual process of conducting an optical microscopy experiment[75].

The approach applied here is also readily adapted to new defect types and systems. The present model was trained with only a relatively small amount of training data due to the use of transfer learning[67]. With only modest additional data sets (e.g., on the scale of thousands of defects or possibly fewer) and a few rounds of further training, researchers could likely extend the present model to more defects (e.g., separating the two orientations of 111 loops or adding voids, preexisting dislocations, etc.), different imaging conditions (e.g., changes in microscopes, imaging modes, orientation, focus, etc.), and different materials (e.g. other metal alloys).

There are several areas where significant improvements may be obtainable. The first is that the use of real-world data in the study has led to significant time spent labeling and introducing unavoidable human biases and errors into the deep learning model and its assessment. However, it is possible that simulated images could be both more accurately labeled and generated in large volume, potentially allowing much more accurate models to be trained.

The second area where significant improvement is likely is that deep learning methods for object detection continue to evolve rapidly. In particular, deep learning segmentation

models[57], which learn a label for every pixel, could be equally or more accurate and remove the step of fitting contours in a bounding box to get geometric information. Such an approach was applied recently to automatically detect information about dislocation lines, precipitates and voids in STEM images[33].

3.2.5 Summary

This study demonstrated a practical deep learning based automatic STEM image defect detection system implemented by incorporating Faster R-CNN for detection and watershed flood algorithm for geometry fitting. Compared with other models proposed before, our model reduced the training effort by utilizing only one module for detection and expanded capability to simultaneously recognize multiple classes of defects. The approach developed here achieved reasonably reliable performance, with an F1 score of 0.78, and predicted sizes and areal densities within the uncertainty of results from human researchers. The automated analysis on NVIDIA's GeForce GTX 1080 GPU processor is about 0.1 s/image, hundreds of times faster than human analysis (≥1 minute/image), and trivially parallelizable and scalable on more processors. The model can also be readily extended to new defects, systems, and conditions with modest training requirements. Thus, our approach provides an accurate, efficient, reproducible, scalable, and extensible method which could replace or greatly enhance human analysis in future studies related to STEM images.

We believe that this framework can be used on many defects and other STEM features simultaneously, eventually providing a general tool for automated analysis across many STEM applications. This study demonstrated a practical deep learning based automatic STEM image defect detection system implemented by incorporating Faster R-CNN for detection and

watershed flood algorithm for geometry fitting. Compared with other models proposed before, our model reduced the training effort by utilizing only one module for detection and expanded capability to simultaneously recognize multiple classes of defects. The approach developed here achieved reasonably reliable performance, with an F1 score of 0.78, and predicted sizes and areal densities within the uncertainty of results from human researchers. The automated analysis on NVIDIA's GeForce GTX 1080 GPU processor is about 0.1 s/image, hundreds of times faster than human analysis (≥1 minute/image), and trivially parallelizable and scalable on more processors. The model can also be readily extended to new defects, systems, and conditions with modest training requirements. Thus, our approach provides an accurate, efficient, reproducible, scalable, and extensible method which could replace or greatly enhance human analysis in future studies related to STEM images.

We believe that this framework can be used on many defects and other STEM features simultaneously, eventually providing a general tool for automated analysis across many STEM applications.

3.3 Defect Analysis of STEM/TEM Videos

3.3.1 Introduction

Transmission Electron Microscope (TEM) has widely been used to characterize a material or material system since TEM provides resolution limits at or below common microstructural features of interest. Recently, a surge in the use of in-situ TEM techniques has occurred, partially due to the advent of digital capture devices. In-situ TEM experiments have a distinct advantage over ex-situ experiments as they allow researchers to study materials' intrinsic properties and responses as external conditions are manipulated such as temperature, pressure, and type of

reaction cells[1]. In material science, in-situ TEM is frequently used to shed light on challenging problems like elucidating mechanisms for catalysis, atomic behavior during material reactions, and nanoscale property changes under loads[76,77].

The value extracted from an in-situ TEM experiment requires careful analysis of the observed processes. For many of these experiments, this analysis includes dynamically detecting features present in the microstructure and analyzing the microstructural evolution in each frame of the experiment, typically captured in a digital video form. For decades, quantification of defects in in-situ TEM data has been completed by humans, which is tedious, time-intensive, error-prone, biased, and impractical to scale. For example, a typical manual workflow of counting defects in TEM images requires an experienced researcher carefully going through every frame for different types of defects and labeling objects in the images one by one. Such manual analysis typically takes many minutes per frame (e.g., in this study, we found it takes about 20 to 60 minutes to process 1 frame, depending on the complexity of the TEM images). Typical in-situ TEM experiments can generate tens to hundreds of frames of video data per minute, so a long video can rapidly become impractical to analyze. Moreover, the labeling quality also depends on the attentiveness of researchers, which may be reduced after spending hours on this repetitive work. Furthermore, other factors such as researchers' proficiency and personal preference when analyzing TEM images contribute to inaccurate or at least inconsistent labeling. Human interpretations are often required for analyzing TEM images that the same researcher may give different labeling results at different times. The above observations imply that manual counting and analyzing methods are hard to scale and prone to human-based errors. In the future, the demand for better TEM analysis methods will only grow, as recent advances in

TEM equipment, e.g., high-speed cameras, and fast microprocessors will keep accelerating the rate of data acquisition[71].

Automatic analysis of TEM/STEM data, especially identifying microstructural defects, is a long-standing pursuit of both the academic community and the industrial sector. To automatically analyze defects contained in TEM/STEM images, various methods have been applied, such as matching key-points in different regions of interest[78], applying different thresholding values to segment different defects[79,80], representing the texture of various targeted structures by the bag of visual words (BoW)[81] or synthesizing artificial image dataset[82], and using traditional machine learning methods, e.g., k-means clustering, to find defects contained in TEM/STEM images[71]. To the best of our knowledge, these methods are only semi-automatic in that they still require extensive human knowledge and time to apply to a given system, and each new material system requires a significant new investment to find an effective approach. Recently, modern deep learning methods have been applied to solve the defect identification problem in static TEM/STEM image data[31–33], suggesting this is a promising approach that could be reliable and highly flexible across many materials and problems. However, deep learning approaches have not yet been applied to the problem of automatic detection and analysis of defects contained in in-situ TEM video. Imaging-related research around TEM/STEM video processing is very active, but has focused on other areas, such as structure reconstruction[83], image quality improvement[84,85], and video alignment[86].

In this work, we focus on the specific task of adapting the deep learning based YOLOv3 model into an automated framework for analyzing in-situ TEM video data. Specifically, we focus on the problem of detection and analysis of radiation-induced dislocation loops generated by an in-situ ion irradiation TEM device, the Intermediate Voltage Electron Microscope (IVEM)

housed at Argonne National Laboratory (ANL). This in-situ ion irradiation TEM device introduces controlled ion beams into a TEM to achieve a high number of atomic displacements per atom (dpa) to mimic the irradiation environment a material will experience in nuclear reactors, satellites, or space stations[87]. The device enables co-irradiation and observation of radiation-induced defects using diffraction-based contrast while the material is being irradiated.

3.3.2 Methods

The in-situ ion irradiation TEM video-based data used within this study has been previously studied and analyzed using the common typical human analysis method[87]. Extensive details on experimental design, human analysis, and resulting materials effects have been previously published[87]. Here, we only present the most pertinent details for context. We selected one of the four model samples (Fe-18Cr-3Al) from the previous study for the current study, but the YOLO-based methods can be easily generalized to other samples or different material systems. The Fe-18Cr-3Al in-situ ion irradiation TEM video-based dataset was generated by performing in-situ irradiation using the IVEM-Tandem Facility at ANL with a pre-thinned TEM specimen titled to the **g**=011 strong two-beam conditions with a frame rate of 15 frames per second using a Gatan 622 video camera. The irradiation was performed using 1 MeV Kr^{++} ions up to 2.5 dpa at a constant temperature of 320°C and a dose rate of $8.3x10^{-4}$ dpa/s. Note, dpa is a measure of the damage or energy imparted into the system and it is known that ion bombardment at the dpa ranges observed generate embrittling defects in Fe-based alloys[53,88]. Under these radiation conditions, it was expected that two dislocation loop types would form, one with a Burgers vector of $a/2\langle 111\rangle$ and the other with $a\langle 100\rangle$ [87]. Under the strong bright-field two-beam condition used where **g**=011 and the deviation parameter, s_g, close to zero, only a fraction of

a/2⟨111⟩ and a⟨100⟩ loops are visible in TEM. To enable direct comparison to the previous human-based analysis where multiple **g**-vectors were used for detection and analysis[55,56], we applied a fractional visibility constant 7/4 to make YOLO detection results comparable with published results using other **g** conditions[87]. Defect size was estimated by assuming the defects are elliptical and the defect size is the length of major axes of the ellipse. The video image size is 1344 pixel x 962 pixel with 2.6884 pixel/nm conversion factor which gives the physical size as 500.0 nm x 357.8 nm. The video consisted of 1175 frames which were linearly related to dpa and time through **Equation 1** which means each frame corresponds to 1.75 seconds.

$$Time\ (s) = \frac{dpa}{8 \times 10^{-4}}$$

$$dpa = 0.8534 + \frac{[(Frame\ Number) \times 1.6466]}{1175} = 0.8534 + (Frame\ Number) \times 0.00140$$

Equation 1

The video data was acquired via frame-by-frame image registration to eliminate sample drift and relevant camera movement. Since there is no landmark frame or feature that can be used to align the video across the whole irradiation dose range, the video was divided into smaller batches for primary image registration, and the final sets from the previous batch were used to carry over the alignment[86,89]. The alignment was done by frame registration based on the selected landmark frame with a template matching and slice alignment plugin[90].

We opted not to use any previous data labeling and decided to label the data ourselves for this project to establish the ground truth data. This choice was because we did not have the exact pixel positions of each defect in the previous study by Haley, et al[87]. We followed the labeling process that has been used in other studies[32]. The ground truth data was labeled by two trained researchers and they checked each other's labeling and explanations for 3 frames before labeling

the real data via an open-source software called ImageJ[60]. Their labeling will be treated as ground truth in this study.

The YOLOv3 model was adopted from an open GitHub repository (https://github.com/qqwweee/keras-yolo3). To train the model, we first converted the pretrained darknet53 weight via COCO dataset into Keras format and then modified the final class number to our defect number, which was one class in our case since we treated all a/2⟨111⟩ and a⟨100⟩ loops as the same type of defect. This single class approach was necessary as Burgers vector determination is not possible using only the **g**=011 condition in the video. We then applied the transfer learning technique to fine-tune the model by freezing the first 245 layers of YOLOv3 and training the last 7 layers[67]. The in-situ ion irradiation TEM video data in this study was composed of 1176 frames and 21 frames were selected and labeled. The sampling was done at random except an effort was made to assure that the sampled frames were approximately uniformly distributed throughout the full set. Among the sampled 21 frames, 15 frames were used for training (trained on 12 frames and validated on 3 frames) and 6 frames were used for testing, where these 6 frames were not seen by the YOLO model during training. The model was trained on GeForce GTX 1080 for 18300 epoch and the learning rate of Adam optimizer was switched between 10^{-4} and 10^{-5} with batch size equal to 4 and Non-Max Suppression (NMS) IoU equals 0.45 to find the optimal weights. Real-time data augmentation operations, e.g., left-right flip, changing hue, saturation, lightness, were applied for the training dataset to enrich the dataset and enhance the performance of the CNN for variations in defect contrast, size, and morphology in the video data set. Real-time augmentation works by augmenting at each training epoch, generating new augmented images in each epoch.

3.3.3 Results

We first tested the performance of the trained YOLO model qualitatively by comparing the detection result of testing frames to that of the ground truth labeling visually, as shown in **Figure 3.3.1**. In general, the automated machine learning program labeled results agreed with the ground-truth labeling by humans, except for certain ambiguous grey spots and when there existed several touching adjacent loops. A zoomed-in comparison between the ground truth labeling and YOLO predictions is shown in **Figure 3.3.2**. The model was also run on the whole in-situ ion irradiation TEM video. In general, the YOLO model successfully detected nearly all the dislocation loops.

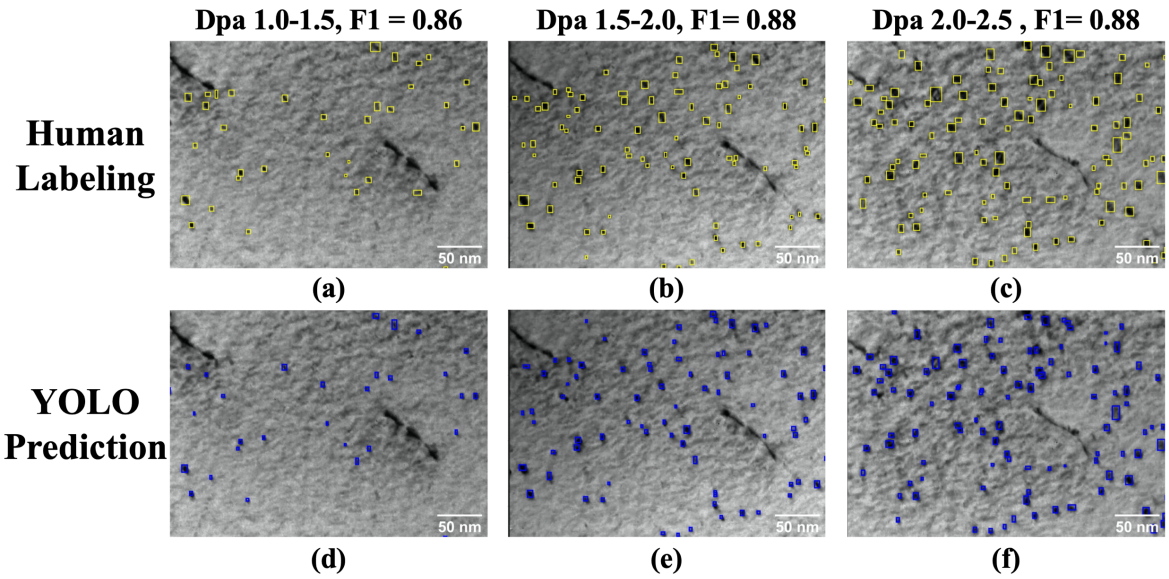

Figure 3.3.1. Selected images from the test dataset for various damage doses (e.g. time scale). Subfigure (a), (b), and (c) are the ground truth labeling developed by two researchers, while subfigure (d), (e), and (f) are labeled by the automated machine learning program. Here, (a) and (d) are for frame number 120, (b) and (e) are for frame number 472, and (c) and (d) are for frame

number 824. 1 frame increment equates to about 0.00140 dpa, see **Equation 1**. F1 score compares the machine detection results with human labeling of each column separately.

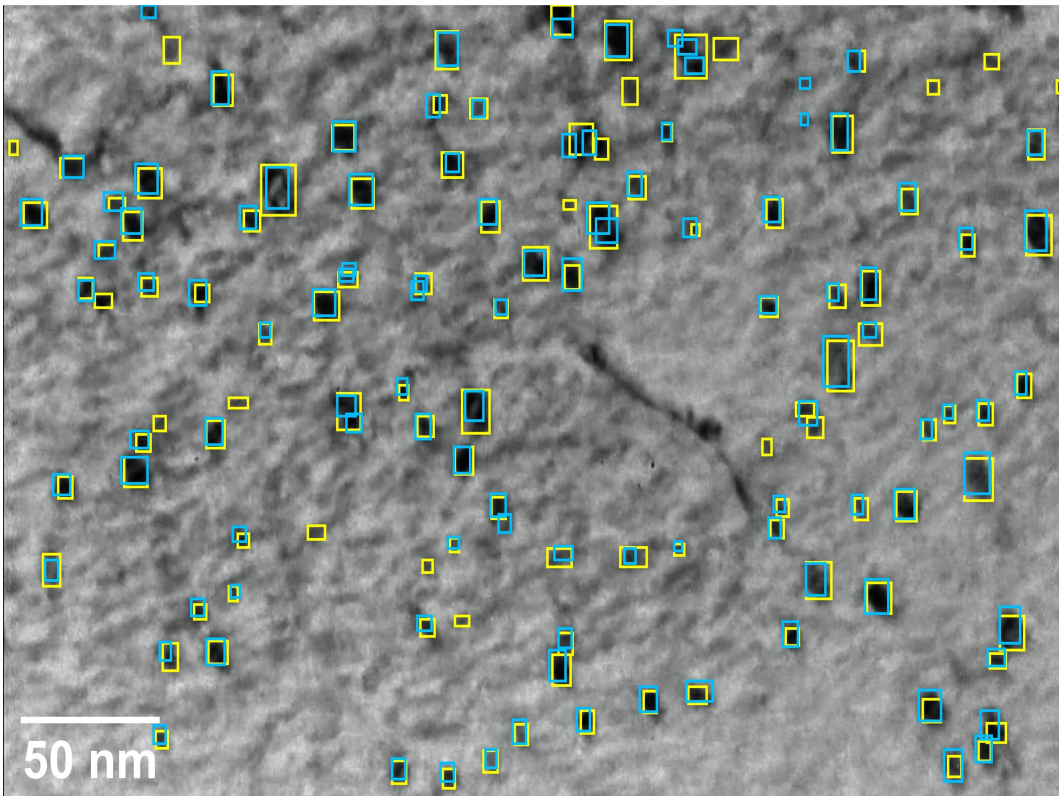

Figure 3.3.2. The visualized comparison of the human labeling results (blue boxes) to the YOLO detector results (yellow boxes) for frame number 824.

The initial qualitative comparison was encouraging so additional quantitative analysis was conducted. The statistics of the model performance were examined based on the metrics of precision and recall and their harmonic mean which is also called the F1 score. The precision, recall, and F1 score were generated using the six test images that were never used in the training process. The test was iterated with different cut-off Intersection-over-Union (cut-off IoU) values as shown in **Figure 3.3.3**. Here and elsewhere in the paper the IoU refers to the ratio of the area of overlap (intersection) to the combined areas (union) of predicted and ground truth bounding

boxes. The cut-off IoU refers to the threshold above which a predicted bounding box is considered as a candidate match for a ground truth bounding box. Predicted bounding box matches are assigned by building a matrix of all IoU values between all predicted and ground truth bounding boxes and making assignments between predicted and ground truth defects using the highest IoU in the whole matrix. When an assignment is made all the matrix entries associated with those predicted and ground truth bounding boxes are removed from the matrix and the process is repeated. This approach provides a unique assignment and effectively assigns the highest overlapping predictions to the appropriate ground truth boxes. In general, a lower cut-off IoU means higher tolerance on the discrepancy between the machine labeled region and the human-labeled region, which agrees with the trend shown in **Figure 3.3.3**, indicating that the performance of the trained model increased as the cut-off IoU decreased.

We selected a cut-off IoU = 0.15 to assess the performance of our model. This value is lower than usually used in machine learning classification problems, but we believe is reasonable for the following reasons. Many defects are small so a shift of just a few pixels in the size and/or center of the ellipse can lead to significantly reduced overlap in bounding boxes. Such shifts are likely within the realm of the uncertainty of human labelers, and of course, the YOLO algorithm makes some location errors, so relatively small cut-off IoU can occur even when two bounding boxes are clearly finding the same defect ellipse. Furthermore, from the density calculation showed below, defects are typically much farther away than their size, with a typical distance at the 2.5 dpa (where defect density is $3\text{x}10^{16}$ cm^{-3}) of about $\frac{1}{\sqrt[3]{3 \times 10^{16}\text{cm}^{-3}}} = 32$ nm. This separation length scale makes it unlikely that boxes of sizes ~ 6-10 nm (the median defect size) on a side will be assigned to the wrong defect just due to allowing a modest overlap.

The F1 score obtained for the cut-off IoU = 0.15 is very encouraging as scores in the range of 0.85 to 0.95 are typically considered very good for object detection results[91,92]. Furthermore, at this cut-off IoU, our 6 testing images are all reasonably accurately modeled, with F1 scores ranging from 0.83 to 0.93.

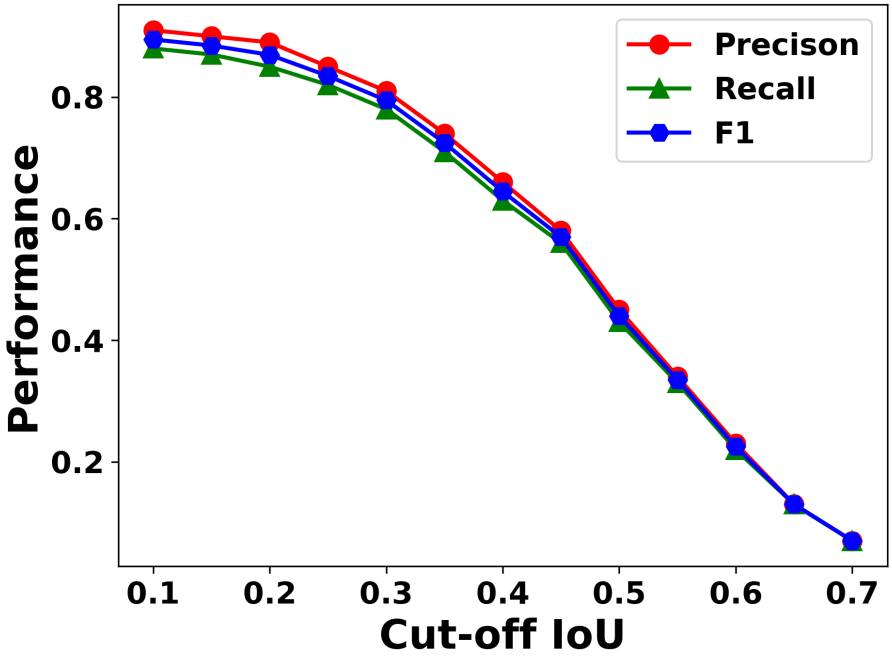

Figure 3.3.3. The performance of the YOLO detector with different cut-off IoU thresholds.

The developed YOLO model was run on each frame of the in-situ TEM video to extract geometry information of each visible defect for the duration of the experiment. After obtaining the geometry and position of each defect per frame, we used this information to extract defect properties, such as median size and number density. Such properties of materials are widely used in the nuclear materials field from which the dataset originated and provide insights into the interplay between the imparted damage and the change in microstructure. We picked four typical frames and compared the machine learning prediction results with ground-truth labeling. Those 4

frames were not used in training and testing the machine learning model. We first compared defect density. Defect density is important for many materials properties and for nuclear materials as it is strongly correlated to mechanical properties, e.g., through the dispersed barrier model[53,93]. Loop density comparisons between machine learning results and our labeling results are summarized in **Figure 3.3.4**. The densities of defects per frame were determined via machine learning (ML) method and manual labeling for ground truth by dividing the total number of the loops by the volume of the sample for each frame. The sample was treated as a rectangle bulk with dimensions of 416.6 x 264 x 75 nm^3 and both results were corrected based on the loop invisibility for the given imaging condition.

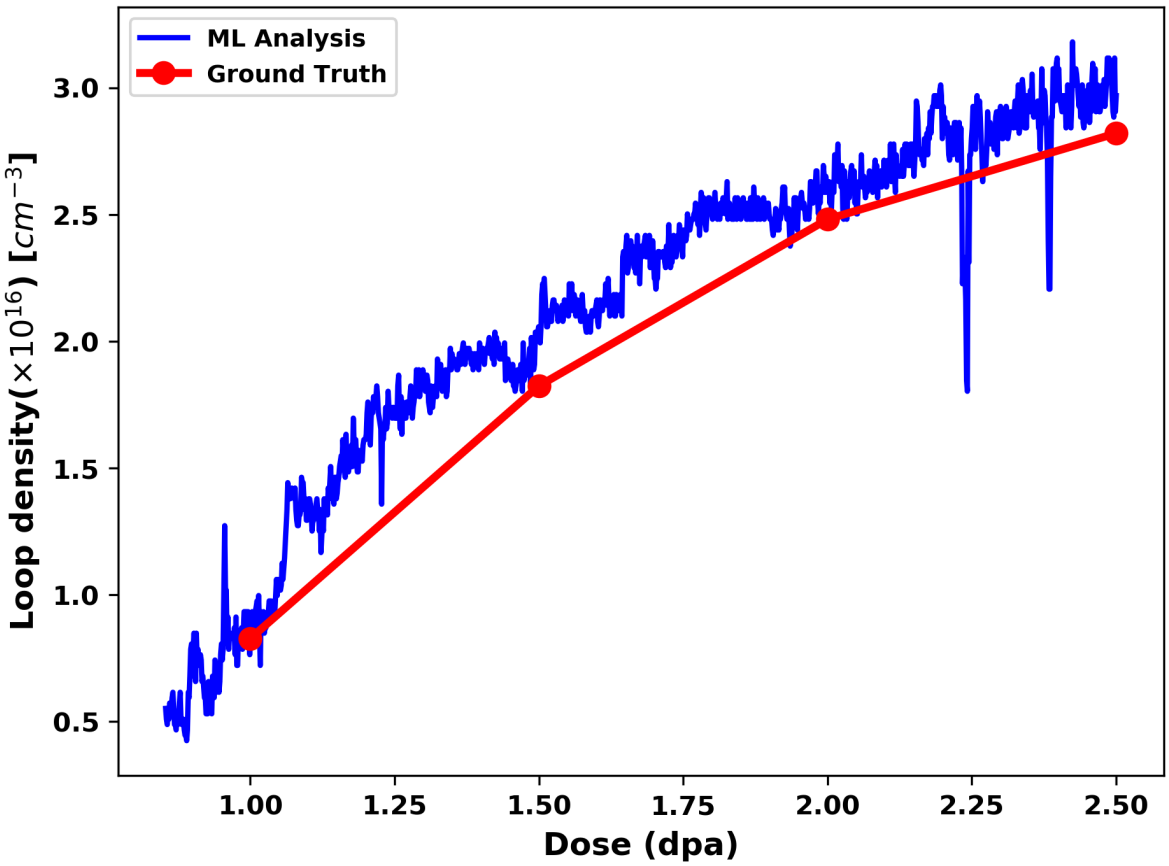

Figure 3.3.4. Loop number density from the whole TEM video. The plot compares the loop number density obtained from the ground truth labeling done by experts in this study and the result obtained from the machine learning detector. All the data shown on the plot uses the corrected proposed density, which is 7/4 of the raw density (see Sec. Material and methods). The sudden drops in the late stage are an artifact arising from camera motion.

Both techniques for analyzing the in-situ videos showed a general trend of increasing loop density with irradiation dose (time) which was expected based on general radiation effect theory and previous analysis of the experiment[94]. Overall, machine learning results were close to the ground truth labeling results throughout all frames, varying at most 12% compared with ground truth labeling at the four measured points data in **Figure 3.3.4**. We believe that the

observed discrepancy between machine learning and ground truth data in **Figure 3.3.4** is likely comparable to different researchers' preferences in labeling ambiguous loops and perhaps cannot be significantly improved without more consistent labeling. It is noteworthy that the sudden drops observed in the late stages arise from abrupt stage movements that rapidly alter the field-of-view and momentarily artificially reduce the effective number of loops observed. This effect is similar to camera movements in the traditional sense.

After obtaining the density of defects from the machine learning detector, we then used a watershed fitting method to determine the morphology statistics of defects. Since all images were recorded with metadata to allow for pixel to physical distance conversions, we could predict the geometric information of each detected defect based on the pixels involved in the defect. We used the watershed algorithm provided within OpenCV[95] to determine the defect pixels and their boundary. Watershed is a commonly used image segmentation method, which divides different objects with watershed lines and then, based on the contour found, extracts precise information about the defects' position, size, and orientations[68]. We used OpenCV's marker-based watershed algorithm. This method requires users to initially label pixels according to their belonging to one of two categories, referred to as the "sure object" and "sure background". The sure objects and background were found by applying a thresholding method, specifically Otsu's binarization and Distance Transformations. To remove noise, we use a morphological opening operation with a 3x3 kernel. We followed the official tutorial from OpenCV, and more details can be found there[69]. Watershed found boundaries of defects and backgrounds, but the boundaries were not very smooth. OpenCV's `fitEllipse()` function was called to fit the needed defects and the major axis length of the fitted ellipse was defined as the defect size. Detailed fitting results with a cropped region of interests are provided in the Data and Code Dissemination section. The

machine learning results of the defect size distributions were compared to ground truth labeling in **Figure 3.3.5**. Although differences were observed in defect median size of each frame in **Figure 3.3.5**, investigation indicated that these differences did not exceed 13.0% difference in median size, and the average difference is only 5.5% and the standard deviation of difference is 5.3% across all doses investigated. The exact formula used to calculate these statistics is given in the Supplemental Information (SI) Section 1. These results indicated that a well-trained machine learning based model could be used for loop detection and analysis and achieve human-like performance comparable or better than the large differences that can be expected by manual labelling[32]. The boxplot comparison provided in **Figure 3.3.5** showed the viability of the machine learning results. At the same time, it needs to be emphasized again that the true strength of such a technique lies in its ability to detect defect information for every frame quickly and accurately instead of just focusing on a small subset of frames.

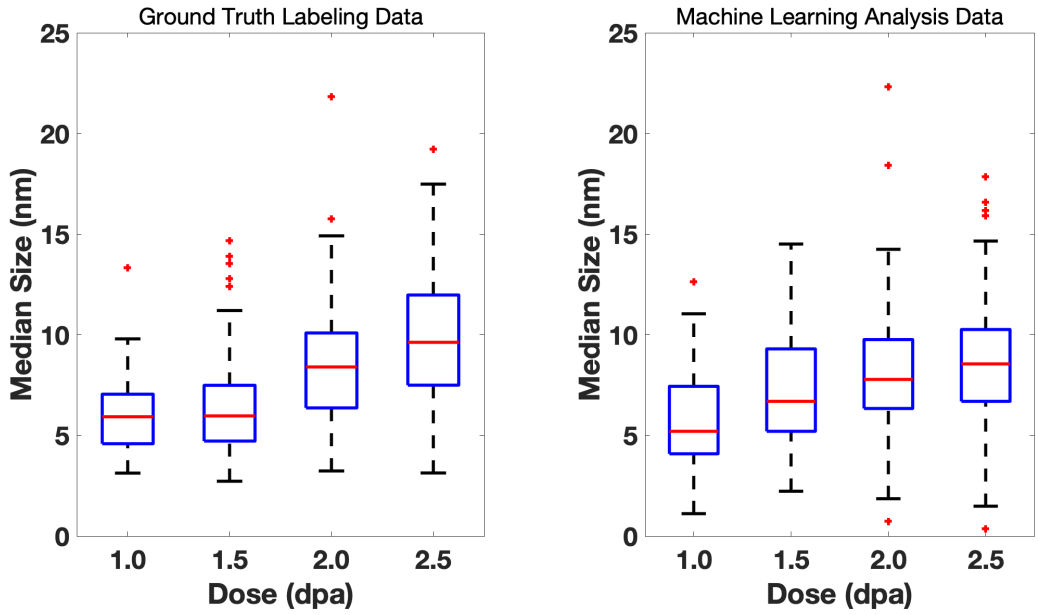

Figure 3.3.5. Box plot comparing the distribution of median size for two methods the ground truth labeling done by experts in this study, and the result from machine learning detector. All

distributions are separately analyzed and compared by their irradiation condition, which is 1.0, 1.5, 2.0, and 2.5 dpa.

Figure 3.3.6 shows the size distribution for the entire duration of the in-situ ion irradiation TEM experiment where, for each frame, the blue line represents the median of loop size, the top of the gray boundary indicates the third quartile of loop size distribution and the bottom of the gray boundary indicates the first quartile of loop size distribution. With the YOLO-based machine learning detector, we could extract data generated in every frame and investigate the material properties with hundreds of times more data than previously collected by hand for this data set (the data collected by hand are the red points shown for four different typical dpa values where these four points are not seen in training data, with red lines connecting them as a guide to the eye). The large amount of analyzed data makes subtle trends easy to identify. For example, although there are some noises, a clear trend can be seen in **Figure 3.3.6** that the median size, Q1, and Q3 increased as the dose value increased from 0.83 to 2.3 dpa and remained stable from 2.3 to 2.5 dpa. Such a result agreed with the relationship found in Haley et al.[87].

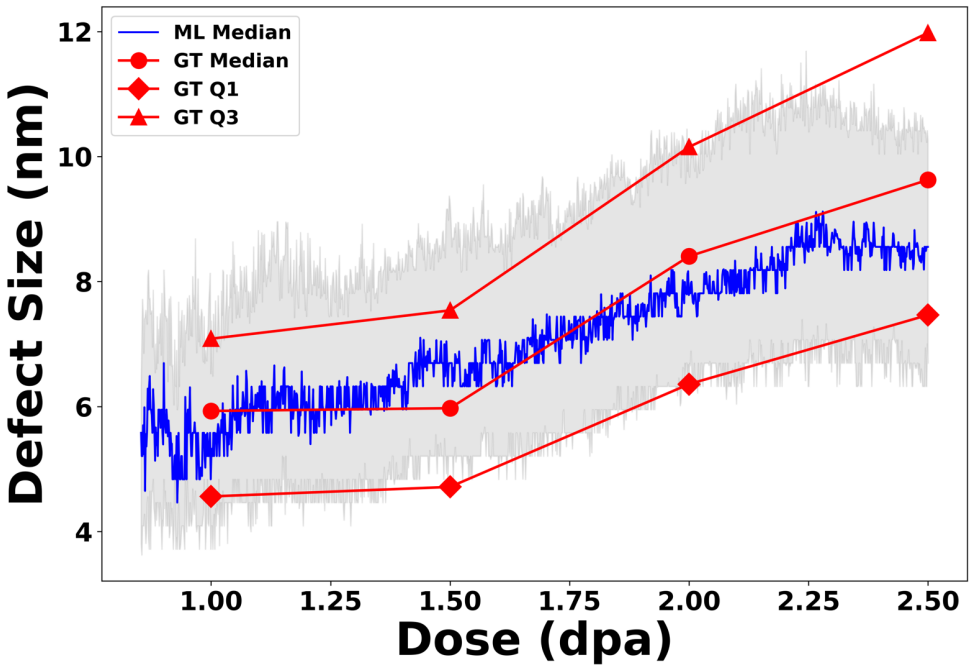

Figure 3.3.6. Change in size distribution as a function of irradiation dose based on machine learning detection. It can be found that the median size (Blue line), first quartile Q1 (Upper gray boundary), and third quartile Q3 (Lower gray boundary) increase as the dose increases when the dose is from 0.83 to 2.3 dpa. Median size, Q1, and Q3 stabilize above ~2.3 dpa. Red lines connect the red points that represent the ground truth labeling of median size (circle), first quartile Q1 (triangle), and third quartile Q3(diamond) of 4 typical frames to provide a guide to the eye.

One of the most exciting applications enabled by automated data analysis of in-situ TEM data is the ability to track all defects as a function of time (i.e., frame). With this application in mind, we developed a tracking module based on YOLO output to track defect motion in the data. Since video is sequential images in time, we can track defects by counting and measuring their sizes across frames to discover their evolution in morphology and mobility under irradiation. This process is usually called object tracking in computer vision studies and is important for

applications such as surveillance and security systems, traffic monitoring, human-computer interaction, etc[96]. One of the most widely used methods for object tracking is tracking-by-detection, also called tracking-by-repeated-recognition[96,97]. In this method, tracking is achieved by detecting targets in consecutive image frames with trained object detectors and linking detected objects across frames to generate the tracking results, e.g., trajectory or motion data[97]. We used Trackpy, a Python package for particle tracking, to link the detected objects generated by the machine learning detector. Trackpy implemented the algorithms first developed for colloidal particles by John C. Crocker and David G. Grier[98] in Interactive Data Language (IDL) and the algorithm worked well for both non-interacting and interacting systems[99]. Trackpy is widely used in the soft matter community for tracking the movement of particle-like objects e.g., colloidal particles or cells in microscopy videos or images. A typical workflow of Trackpy can be split into three steps: (1) Locating Particles, (2) Refining Location Estimates, and (3) Linking Locations into Trajectories[98]. In the first step general features of particles like diameter, maximum size, and separations are used to locate all peaks of brightness in the image which includes the initial object coordinates. Subsequently, more pixel-level information is used to distinguish real particles from spurious ones. Finally, the locations of particles in each image are matched with corresponding locations in later images to yield the whole trajectories. The tracking module is a powerful tool to obtain several important statistics relating to the motion and evolution of defects. When combined with automatic labeling, it provides a new way to study defect dynamics under irradiation at a fidelity not possible using previous methods. We demonstrated this advantage by two case-studies using the tracking algorithms: (i) studying defect evolution and trajectory of interesting defects and (ii) extracting statistics of individual defect mobility e.g., diffusion coefficients.

To study defect evolution and trajectory of interesting defects, we first showed the size change of an individual defect, then compared the trajectories of the slowest moving defect and the fastest moving defects, and finally, showed the landscape of defect moving trajectories. In **Figure 3.3.7**, a defect was shown to undergo significant size change as the dose increased. With the help of the tracking module based on the YOLO and Trackpy package, a full history description of a defect's size change was recorded to illustrate the evolution of the defect. The defect size change is shown in **Figure 3.3.8** which clearly indicated that defect size increased as radiation dose (dpa) increased.

Figure 3.3.7 and **Figure 3.3.8** show the ability to extract a single defect growth evolution as part of this in-situ TEM experiment. It is interesting to note that the shape/trend of the growth rate for individual defects varied, with some showing unconstrained linear growth and others showing asymptotic growth, and even some showing growth followed by shrinkage. Although not the focus of this study, we believe the different growth curves for individual defects could be attributed to local variation in the direct vicinity of the defect, and these variations could promote or retard growth under irradiation. Significantly more analysis of the data would be required to evaluate the postulated mechanism. But even at the level of the analysis presented here, the power of such individual defect tracking is obvious.

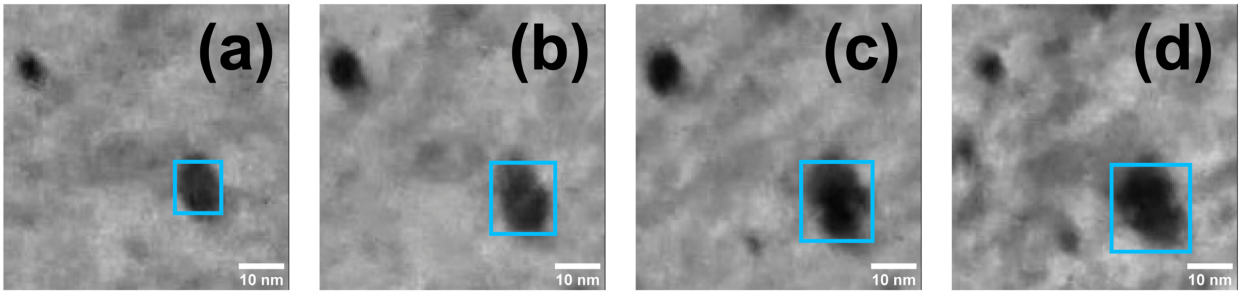

Figure 3.3.7. Reduced field-of-view bright-field TEM images of a single dislocation loop growing under increasing irradiation dose of 1.28 displacements per atom (dpa), 1.72 dpa, 1.95 dpa, and 2.35 dpa for a)-d) respectively. The highlighted loop shows the dynamic change in contrast necessary for the tracking model to detect and quantify. The defect id (51) was assigned by Trackpy.

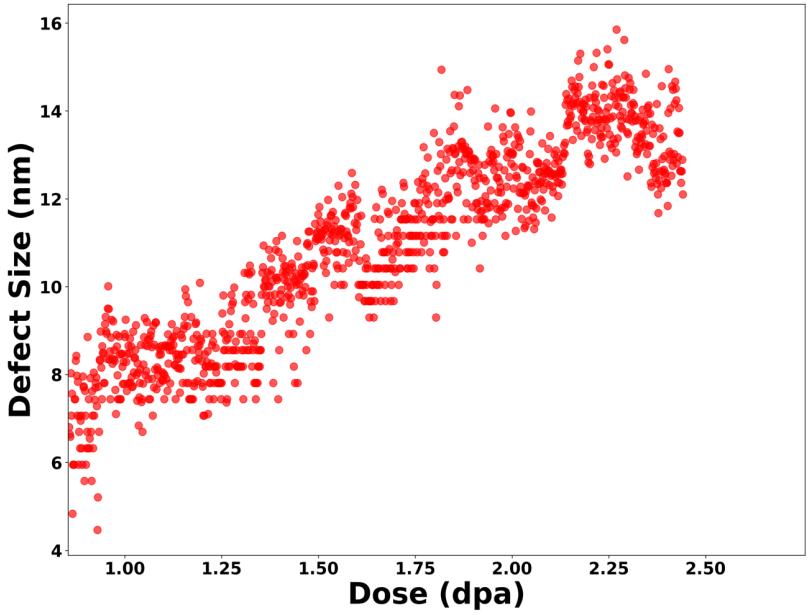

Figure 3.3.8. The size change of a single typical defect, which is the same defect shown in **Figure 3.3.7**.

Since the individual history of every defect was obtained, it was straightforward to examine defects with interesting behaviors. For example, as shown in **Figure 3.3.9**, our tracking

module could determine the motion of very slow-moving (**Figure 3.3.9**(a)) and fast-moving (**Figure 3.3.9**(b)) defects.

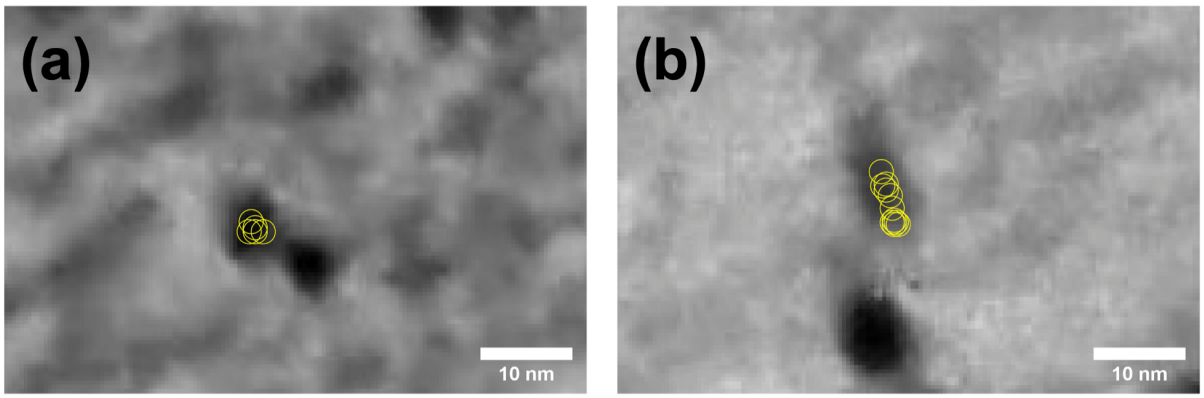

Figure 3.3.9. Trajectories of two typical defects throughout their lifespan. (a) represents a typical defect that has close to minimum diffusion coefficient value. Figure (b) represents the defect that has a nearly maximum diffusion coefficient. Each yellow circle center represents a specific location of the defect in certain frames, and the set of locations are plotted on a single image to show the relative movement. The defect id is assigned by Trackpy.

The spatial distribution of defect trajectories was also an interesting property that was determined and is shown in **Figure 3.3.10**. It is noteworthy that in the original video source, due to thermal expansion of material and TEM user operations under irradiation, the viewable area adjusted somewhat over time. This movement is an artifact of the in-situ experiment, but the Trackpy package corrected for these artificial movements enabling us to target only the real movement of each defect.

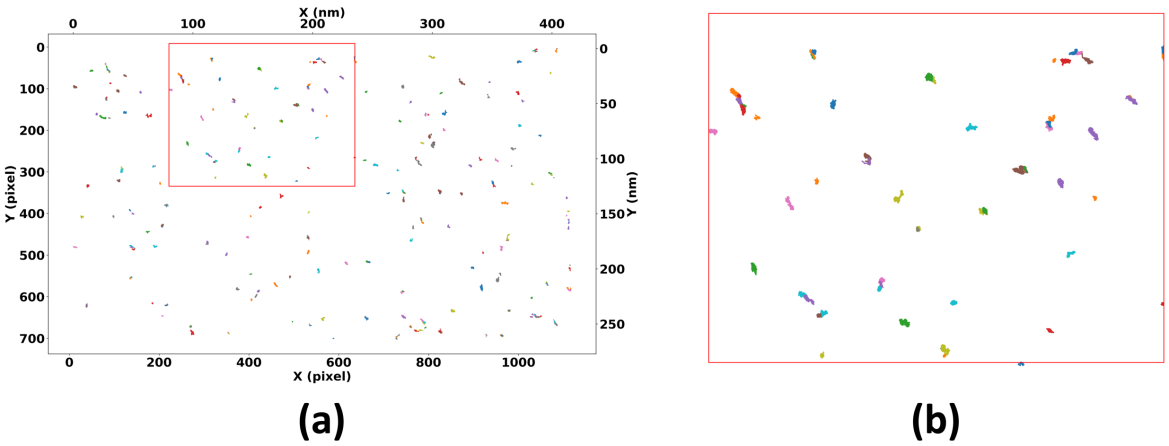

Figure 3.3.10. The trajectory of typical defects detected in TEM video. The movement of this type of defect is roughly cyclic, so the trajectory is not a single line but rather a small group of points. Results were generated by Trackpy. Subfigure (b) is a zoomed-in result of the red rectangle in subfigure (a).

Since we knew each defect's position and time stamp, an effective two-dimensional diffusion coefficient (D_{eff}) can be determined. Diffusion of defects is an important property of defect behavior in nuclear materials[100]. We calculated this effective diffusion coefficient using the following relationship:

$$D_{eff} = \frac{|r(t+\tau) - r(t)|^2}{4 * \tau}$$

Note that D_{eff} is not a true diffusion coefficient as we make no effort to correct for the two-dimensional projection of the three-dimensional defect motion, which can be complicated by the exact angle of the sample and the detailed motion of the defect. Our goal for this work is merely to demonstrate the ability to track trajectories through the combination of YOLO and tracking tools, not to perform detailed analysis to extract physically meaningful diffusion coefficients. To perform the analysis, we choose 345 consecutive frames (from 1176 total) over which the camera appears to be very steady. These frames are from frame numbers 461 to 805,

corresponding to dpa values from 1.50 to 1.98. Only the regions away from the edges of the figure are used to avoid defects appearing and disappearing due to small changes in the image region. Specifically, we consider only the region with Y position from 200 to 1450 pixels and X position from 250 to 2150 pixels where the original size of the image is 1728 pixels in the Y-axis and 2412 pixels in the X-axis. We find a total of 741 defects in Trackpy, which is significantly larger than the number of defects in a given frame. This larger value is due to the fact that defects have a finite lifetime due to their appearing over time and, in some cases, disappearing, which leads to more tracked defects than actual defects in the analysis. Our average lifetime is 54 frames. While some of the defects may actually appear and disappear, many of these events are clearly artifacts due to Trackpy inadvertently assigning multiple global IDs to the same defect, which effectively causes one defect to disappear and another to appear even when it has not actually done so. Such errors make our defect counts inaccurate from TrackPy but do not lead to incorrect estimates of the defect's D_{eff}. To illustrate the values of the diffusion coefficients, in **Figure 3.3.11** we show the distribution of D_{eff} as a function of binned defect median sizes. We used "median defect size" as a defect would have different sizes in each frame where it is identified, either due to small changes in size estimates from the numerical analysis or due to the defect growing during the irradiation. We then calculated the average D_{eff} of defects that fall into the same bin, where we have 50 bins from 2 nm to 18 nm. While this figure illustrates the type of correlation one can explore with the automated data analysis, in this case we find no statistically meaningful trend with defect size.

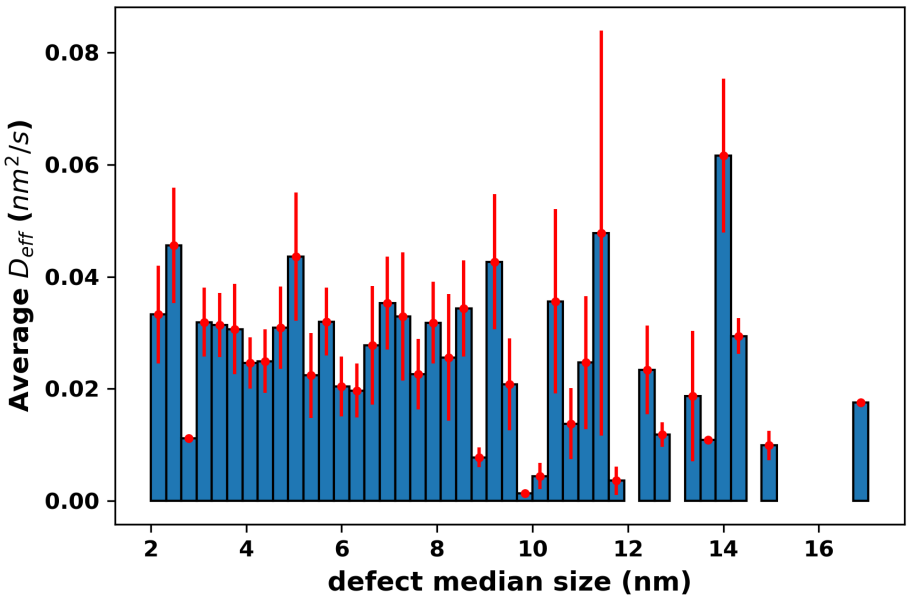

Figure 3.3.11. The distributions of the effective diffusion coefficient Deff calculated by Trackpy as a function of the defect median defect size. The data is presented as a histogram with each bin of width 0.32 nm, giving 50 bins from 2 nm to 18 nm. The height for each bin is the mean Deff of all defects in that bin. Error bars are the standard deviation of the mean. We use "median defect size" as a defect will have different sizes in each frame where it is identified, either due to small changes in size estimates from the numerical analysis or due to the defect growing during the irradiation. No error bars are given for bins with just one defect as the errors cannot be readily estimated.

3.3.4 Discussion

To further validate the detection results generated and analyzed by ML methods we developed in this study, we compared our results with those previously completed by Haley et. al. who investigated the same TEM video with conventional manual analysis method[87]. Based on the comparisons, we concluded that the results generated by our ML method are close to those

determined by human experts. For example, the discrepancy between the ML generated loop density differs from the results in Haley et al. by at most 38%. And the difference can be largely attributed to the researchers' preferences, as different experts may have different labeling preferences for ambiguous objects. Likewise, the difference between statistics of median size distribution from ML and Haley et al. did not exceed 32% difference in the mean size, 24% difference in median size, and 30% in the standard deviation across all doses investigated.

It is important to be sure that the model is robust to at least some reasonable levels of noise. To test the sensitivity of the model we used scikit-image (https://scikit-image.org/docs/dev/api/skimage.util.html#skimage.util.random_noise) to add Poisson, Salt and Pepper, and Gaussian additive noises to the test images. We calculated the precision, recall, and F1 scores from the model for a range up to quite significant added noises and the impact on the performance is less than 20% in the F1 score for all cases. This impact is relatively minor and suggests our model is quite insensitive to noise.

Since the YOLO object detection model performance is lower for very small objects[101], there exists a threshold of defect size below which our model cannot detect a defect. Similarly, there is a defect size below which human labelers do not label a feature as a defect. It is important that the human lower limit is larger than the YOLO lower limit or otherwise we will systematically fail to identify very small defects. The human labeling threshold value was estimated as 7.24 pixels (2.69 nm) based on the lower limit in our labeled data. The YOLO object detection algorithm finds defects as small as 1.86 nm, so YOLO is able to find defects as small as any human chooses to label.

Although the performance of the detector we used was quite accurate for defect recognition in TEM video, improvements to the model are needed. Errors likely could be

reduced by more extensive optimization. For example, in the training, we only used default anchor box settings and K-means clustering of bounding box sizes in training data could be a better way to find the best set of anchor boxes. Also, more data augmentation operations could be applied e.g, rotation, adding noises, and cropping or affine transformation to achieve better performance. Errors could also be reduced by removing biases and ambiguities in the labeling. For example, it was often unclear how to establish the ground truth labeling of closely distributed objects with no significant white space between two centers.

It should also be noted that the images used in this study are of very high quality, with limited noise and few confounding contributions (e.g., surface oxide), and undergo fairly modest changes during the irradiation (e.g., few defects move significant distances). The high-quality and modest changes of the images almost certainly help the model performance and subsequent defect tracking. Furthermore, we focus our model on only one type of defect, a single category of dislocation loop, to reduce the burden of labeling and focus on the most prevalent defects in the images. Many samples will have other types of defects (and some are even present in our images, e.g., dislocation lines), and tracking these is an important area for future study. While there is nothing intrinsically limiting YOLO to just one defect type (YOLO could be extended to 9000 classes of objects[102]), what we studied in this paper is a very simple case. To fully assess the general effectiveness of our approach and develop a broadly applicable tool, the model needs to be demonstrated on many more data sets with multiple defect types, varying image and sample quality, and more complex defect evolution during irradiation. However, the present deep learning model is a powerful proof of principle and suggests that a broader program may be successful and have a major impact on the defect detection community.

For future study directions, we think two major directions are worthy of investment. One is creating high labeling quality data sets. For example, in this study, we combine a/2⟨111⟩ and a⟨100⟩ loops together to alleviate the labeling burden, but it will be more informative if we can differentiate these two types. Such high-quality labeled data does not necessarily have to come from experiments and synthetic data can have many advantages. For example, image simulation, such as the multi-slice method, can generate high-quality images filled with known types of defects[103]. This method can help avoid the tedious, error-prone labeling process. Synthetic images might also be generated with deep learning methods such as Generative Adversarial Networks (GANs)[104,105], which are powerful tools for generating images similar to an existing set. GAN generation might be done in such a way that labeling is automatic, creating an almost unlimited supply of high-quality labeled synthetic images or converting images collected from different conditions to the condition for which our model is trained, allowing the community to better utilize limited labeled data[106–108]. The second direction worth exploring is to apply the analysis system developed in this paper to TEM devices to provide real-time statistics and even direct labeling of defects (e.g., with a fitted ellipse) in images to guide users during experiments. This approach is similar to the real-time Augmented Reality (AR) methods that have proven to be useful in biological microscopy studies[75]. This combination will provide a straightforward, real-time output of deep learning analyzed results for TEM studies and the material community.

3.3.5 Summary

In summary, the present work shows that if the accuracy obtained here can be extended to more general and complex data, these deep learning tools are a potentially transformative methodology for the TEM community. The YOLO based system developed in this study

provides an automatic, fast, and reliable quantitative analysis of both position and morphological evolution of defects in frame level. Furthermore, the YOLO based system can help researchers track the motion of defects, which will allow new levels of dynamical analysis. Furthermore, the approach is easy to use and adapt to other sets of experiments. The speed of YOLO means that it can be used in real time to adjust experimental conditions (e.g., dpa, temperature) or imaged regions (e.g., near grains boundaries vs. inside grains), providing a critical tool to support real-time TEM video analysis for material property exploration. We anticipate this YOLO based analyzing system will significantly enhance the capabilities of in-situ TEM/STEM image analysis.

3.3.6 Data and Code Dissemination

All data and code files are stored in the Materials Data Facility[109,110] at DOI: 10.18126/n9dj-5mk0. They are described in detail below.

- Raw Data: In the folder Raw_Data, we provide the original TIFF format video and the converted 1176 JPG images of each frame and the cropped center region of interest 1176 JPG images.
- Labeled Data: In the folder Training_and_Testing_Dataset, we provide the labeled data and the data is already put into the TRAIN folder and TEST folder. One needs to put the full path to these directories in the YOLO labeling file (called “train.txt” in our codes).
- Code: In the code folder we provide the codes. Specifically, we provide all the codes we used in organized into Test, Train, and Trackpy subfolders of the Code folder, based on their respective applications.

- Fitted Defects Contour Results: We provided the fitted defects of original size videos and cropped region of interest as MP4 videos in the FittedDefects_video folder.
- Plotting Figures and Data: In the folder YOLO_Figures, we provided all the scripts and data we used to plot figures shown in this paper, and subfolders are named by the index of figures.

We also provide all codes in user-friendly IPython notebooks through GitHub at https://github.com/uw-cmg/DefectSTEMVideoAnalysis.

3.4 Mask R-CNN

Both Faster R-CNN based(**Section 3.2**) and YOLO based(**Section 3.3**) defect analysis system is bounding box oriented, meaning the output of the deep learning system is bounding box around defects which is called object detection in computer vision, and the exact geometry information needs to be extracted by pixel segmentation algorithm like watershed. However, Mask R-CNN, an object segmentation model, is able to output object class labels for all pixels following the exact end-to-end style[111]. We labeled the dataset used in **Section 3.2** and **Section 3.3**, and trained the Mask R-CNN model provided in the Detectron2 package which was developed by the Facebook AI Research (FAIR) team[112]. Example output was visualized in **Figure 3.4.1**. We can find Mask R-CNN provides pixel-level class labels and the results agrees with human labeling quite well.

We have submitted this work in October 2021 which is under review now and the Arxiv link is here, https://arxiv.org/abs/2110.08244. Although I prepared the data and ran the model

analysis as part of this work the bulk of it was completed by Dr. Ryan Jacobs, so a detailed analysis of Mask R-CNN performance is not discussed further in this thesis.

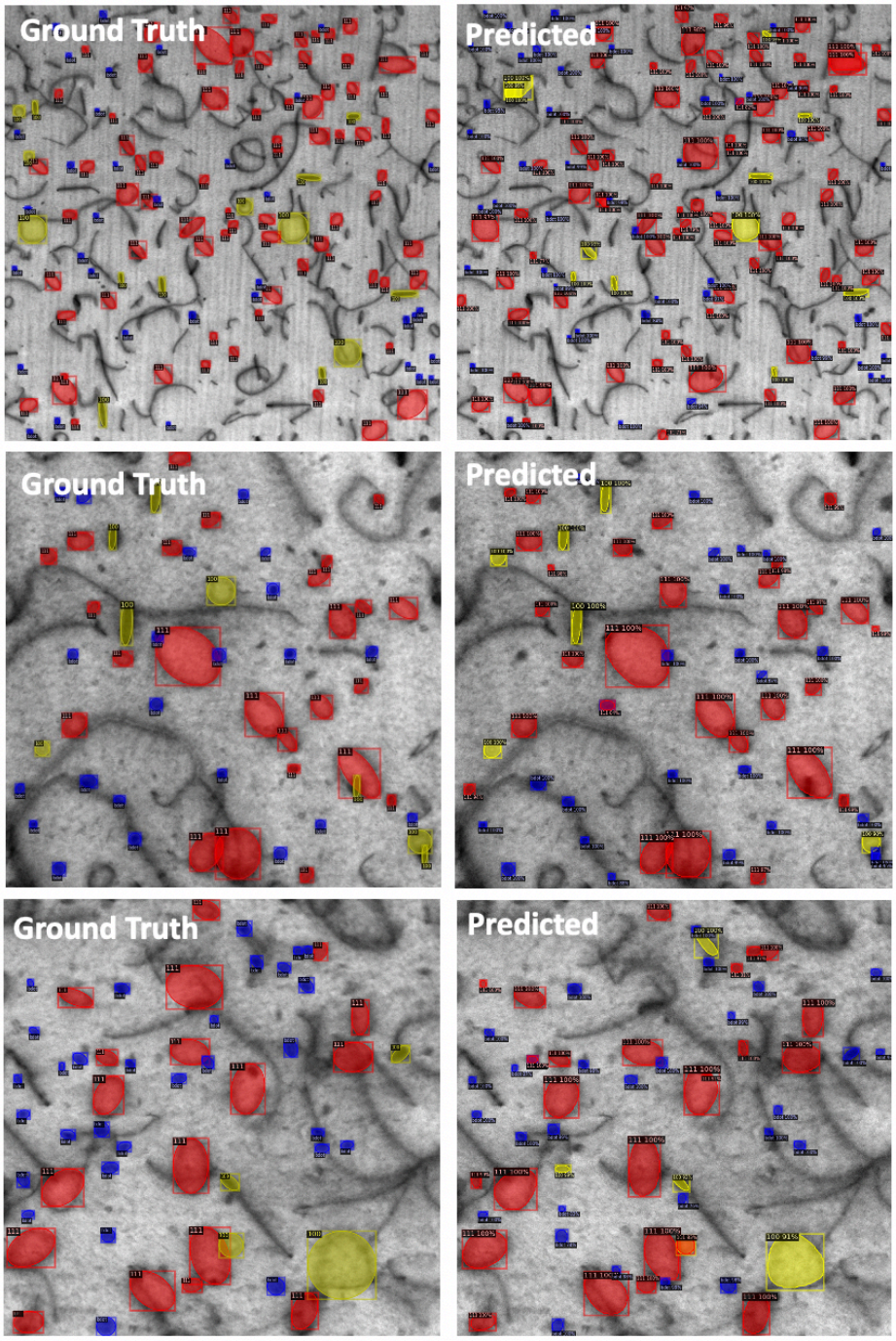

Figure 3.4.1. Examples of labeled ground truth (left columns) and Mask R-CNN predicted (right column) images. The red, yellow, and blue masks denote 111 loops, 100 loops and black dot defects, respectively. The predictions shown here were made with IoU=0.3.

3.5 Conclusions

Three deep learning based defect analysis system has been built and it paves a way for material scientists to apply state-of-the-art object detection and segmentation models to better accommodate the on-going deluge of microscopic images or videos[113].

Chapter 4 : Machine learning for interpreting coherent X-ray imaging patterns

4.1 Chapter Abstract

In this work, we developed a deep learning classifier system that could extract number density from coherent X-ray image patterns and this work presents a prototype work for better understanding the image patterns.

4.2 Introduction

X-ray imaging has been widely used by many researchers in material sciences e.g. investigating micro, nano or even atomics scale structures[114] or mechanisms[115], studying dynamics or correlation behaviors[116–118], revealing nebulous phenomena in complex material systems e.g. complex fluid[119] and metallic glass[120] and other areas e.g. medical imaging[121], geosciences[122]. One of the most important recent advances in X-ray imaging is utilizing coherent x-ray light sources to investigate behavior on the femtosecond time domain and structure on interatomic length scales. Such characterization is enabled by development of X-ray sources such as advanced synchrotron sources, X-ray free electron lasers, and high harmonic generation sources[123,124]. For example, those new X-ray imaging methods can visualize the chemical composition in nanoscale resolution[125], study 3D lattice dynamics in gold nanocrystals[126], and shed light on complex systems like biological system e.g. showing 3D mass density distribution of a whole, cell[127] or reconstructing 3D structure of the giant mimivirus particle from diffraction patterns[128]. However, due to the complexity of interactions between sample and coherent X-rays, the imaging results, called X-ray imaging patterns, are difficult to interpret. It is particularly challenging to extract information about the molecular structure from images, and such reconstructions to date have often relied on

significant approximations and simplified models e.g., using phase retrieval algorithms to iterate between real and reciprocal space[129,130] or alternating projections[131] ,angles or sample positions[132]. It is therefore of interest to ask whether recent developments in image analysis using machine learning might aid in the interpretation of X-ray imaging patterns. In this study, we proposed to build a deep learning system that classifies the coherent X-ray image patterns according to their micro-scale structural information, the total number of disks contained in each image pattern or more specific the disk number density. Although other physical or chemical values might also be of interest, we choose disk number density since it is quick and easy to test e.g., only disk number needs to be counted. To reduce development difficulty and move quickly to build working prototype, we focused on 2D system not the actual 3D real experiment dimension. The classifier was trained in simulated coherent X-ray image patterns without any assumptions or simplification of samples and the results suggested that it was possible to use machine learning tools to corelating coherent X-ray imaging patterns, k-space information, to structural or particle distribution information e.g., disk number density, real-space information which was typically hard to obtain. The method directly builds a mapping between real-space sample information e.g., disk number density and the k-space coherent X-ray imaging.

4.3 Methods

We used forward simulation methods to generate coherent X-ray speckle pattern images of a model 2D disk system and then trained a Convolutional Neural Network (CNN) model called Resnet-50 to classify the X-ray images into different categories corresponding to their disk numbers. The workflow pipeline of this approach is shown in **Figure 4.3.1**.

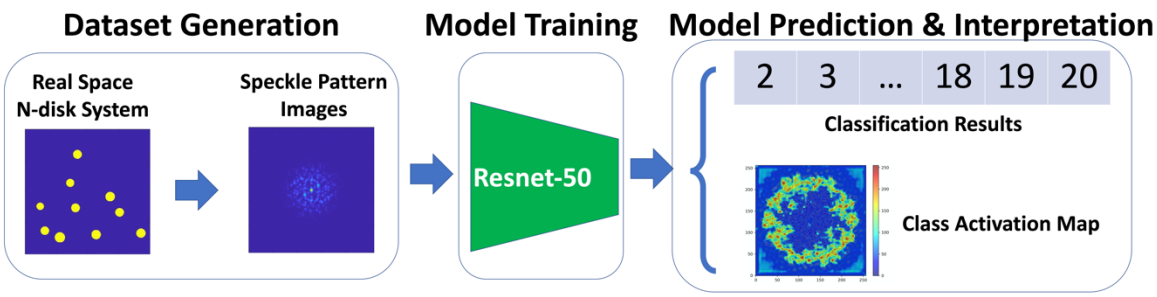

Figure 4.3.1. Flow chart of machine learning system which includes dataset generation, model training and model prediction and interpretation. The generated speckle pattern images were fed into the Resnet-50 classifier system and then classification results of the speckle pattern based on real-space number of disks and the corresponding class activation map to indict which region of the speckle pattern image contributes most to the final classification of speckle pattern images.

The 2D system used in our study was 2000 x 2000 in reduced unit with respect to define 1 pixel as unit in k-space and the total area of disks is $2 \times \pi \times 90 \times 90$ where 90 is radius of the two-disk system. For an n-disk system the radius of each disk was set to preserve the total area of disks, which requires $r = \sqrt{\frac{2\times90\times90}{n}} = \sqrt{\frac{16200}{n}}$. Then the n disks were randomly placed into the 2D system without overlap of disks. For a select set of studies focused on polydispersity which further modified the disk radii to create a polydisperse disk system. To create a polydisperse disk system, the single radius system of $\underline{n}$ disks was modified by sampling the radii from a Gaussian distribution of given mean (3 in our settings) and standard deviation (1 in our settings) with the constraint that the total area of disks is kept the same as the original single radius n-disk system. The given mean and standard deviation were chosen arbitrarily. To achieve the poly-dispersed distribution, we needed to sample disk radii from Gaussian distribution with the same total n-disk area constraint. We assume the n disks radii have a common factor called k and each radius could be expressed as kC_i, so the total n-disk area constraint could be written as

$k(C_1 + C_2 + C_3 + \cdots\cdots + C_n) = 16200$ where C_i is sampled from the Gaussian distribution (with negative values rejected) and k is the common factor needed to be calculated by summarizing the n sampled C_i and once k is obtained, the radius of n disks can be gotten using the equation $r_i = \sqrt{C_i \times k}$.

To generate X-ray imaging pattern data, we use a Fourier Transform program to covert the 2D n-disk system of real space to the k space images. Since the k space image could be infinite, we truncate the image into size of 513x513 pixels. In **Figure 4.3.2**, we show real space and corresponding k space images for select n-disk systems (n = 2, 3, 6, 10, 15, 20). In total we consider 19 cases given by n = 2-20.

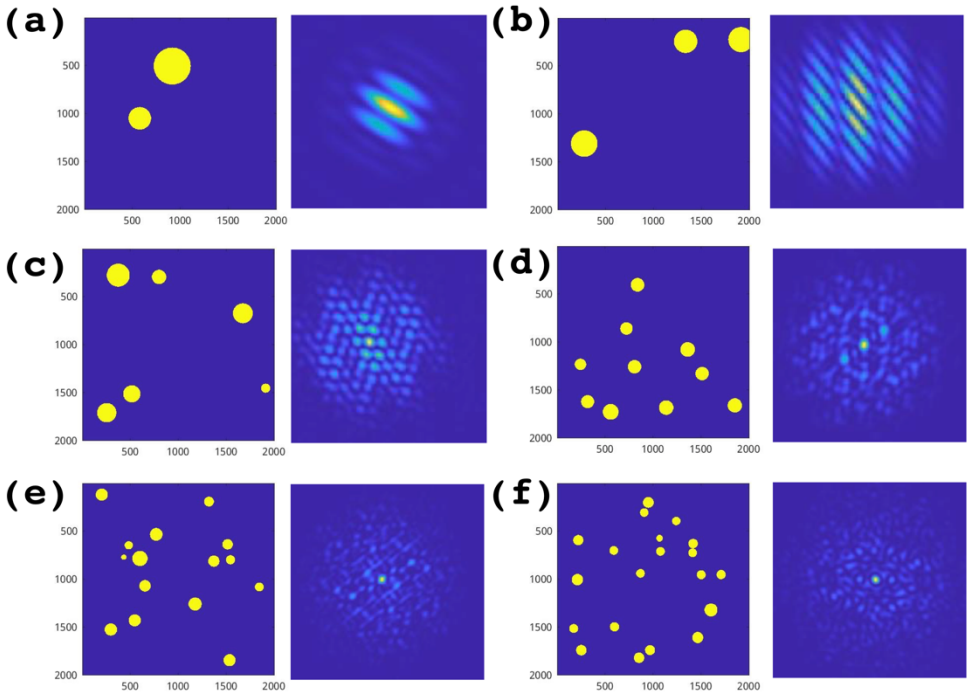

Figure 4.3.2. Real space images and corresponding X-ray speckle pattern, arbitrarily tailored to highlight the speckle patterns, images of six typical 2D disk system where (a) for 2-disk system,

(b) for 3-disk system, (c) for 6-disk system, (d) for 10-disk system, (e) for 15-disk system, and (f) for 20-disk system.

After getting the X-ray speckle pattern images, some preprocessing was needed to delete easy hints about disk number before feed into the machine learning system as we want the machine learning system to find some intrinsic and hidden pattern in the speckle pattern images not to find easy hints to take a shortcut to classify X-ray speckle pattern images. For example, the pixel of and around the center pixel is and is near the Fourier transformer of $q = 0$ competent which is a direct reflection of the area of scatters which may reveal the number of disks in the system, so we blocked the 11 pixels times 11 pixels center region of X-ray image speckle pattern to avoid this easy hint. This is also a common practice used by experimentalists in real world since the center pixel is too bright and hence makes it harder for detectors to better track X-ray patterns[132]. And standardization was applied to each image to make sure the pixel intensity of all speckle pattern images should have the same zero mean and unit variance.

In this work, we proposed to use the widely used CNN model called ResNet-50 to classify the speckle pattern images based on the number of disks. ResNet[133,134] is a family of deep learning model that uses identity mappings to overcome the performance degradation problem of stacking more layers[135]. It is the first deep learning model that achieved lower than human level error rate in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2015 competition[24]. For this work, we created each 1000 different random configurations of disk position for 19 different disk number systems. In total, we have 19,000 X-ray speckle pattern images and among them, 15200 images were used for training, 1900 images were used for validation, and the remaining 1900 images were used for testing. And for the 1900 images, each

disk number system has 100 images which were randomly selected from the 1000 total images of that disk number system. And for training and validation set were randomly split from the left 17100 images. ResNet-50 model was written in Keras[136] with TensorFlow[137] as the backend engine and the training was using Adam optimizer with default setting of Keras and typical training was conducted 200 epochs with batch size of 40 unless early stopped if the changes were smaller than 0.00001 for 40 epochs.

To fully understanding the working of the X-ray imaging classifier system, we introduced the Gradient-weighted Class Activation Mapping (Grad-CAM) as a visual tool to help diagnose the classifier results. Grad-CAM is a technique to generate 'visual explanations' for Convolutional Neural Network (CNN)-based classification models by utilizing the gradients of any specific class with respect to penultimate (pre Dense layer) Conv layer output to produce the high-resolution class-discriminative localization map that highlights the important regions for final classification decisions[138]. Grad-CAM is one variant of Class Activation Mapping (CAM)[139] tools that is widely used in interpreting CNN based classifier systems[140–148] and Grad-CAM is able to be applied to a wide variety of CNN based models[138]. We used Grad-CAM function provided by Keras-Vis package to generate Grad-CAM maps of X-ray imaging classifier systems[149].

4.4 Results

We first present the non-polydispersity system results, e.g., all beads were with the same sizes and then we show the polydisperse system results, e.g., all bead sizes were following a Gaussian distribution (the construction of the polydisperse system was discussed in **Section 4.3**).

4.4.1 Non-Polydispersity System Results

We showed the non-polydispersity classification results of 19 different disk number in **Figure 4.4.1.1**. **Figure 4.4.1.1** demonstrates that the classification algorithm works well, and misclassification only happens once in beads number 19 and only shows a small error of 17 instead of 19. This one error is the only misclassification in 1900 testing cases. The cross-class accuracy is nearly 99.9% which demonstrates the capacity of Resnet-50 based deep learning classification system for coherent X-ray imaging patterns.

To better understand the mechanism by which the machine learned network identifies disk number from X-ray diffraction image we plotted the Grad class-activation map (**Figure 4.4.1.2**). For space consideration we only showed the even bead number cases in the text and the remaining Grad-CAM examples can be found in the Supplemental Information (SI). Although the specific rules used by the deep learning classifier remain unclear, we found that in this problem Grad-CAM could aid the interpretation of classifying results by indicating the highlighted regions of X-ray image patterns or features that could assist human understanding e.g., for certain classes center region seems more useful than conners.

In examples above, we focused on 257x257 pixel size as the coherent X-ray imaging inputs, and it is important to know that whether different size of inputs will change the classification results. Since k space could be extended to infinite, so truncations were needed, and different size of image sizes was setting a limit of real space signals e.g., signals caused by small distances would be filtered out. Hence it was important to know, how the classifier system works in different frequency ranges. We showed the confusion matrix of 513x513 pixel size inputs in **Figure 4.4.1.3** which shows little changes compared to **Figure 4.4.1.1**. The drop of

performance is very small which indicates 257x257 is enough for our classification problems even though larger size of inputs has more information.

In summary, the deep learning classifier works for identifying the number of disks from coherent X-ray imaging patterns in our model non-polydispersity systems.

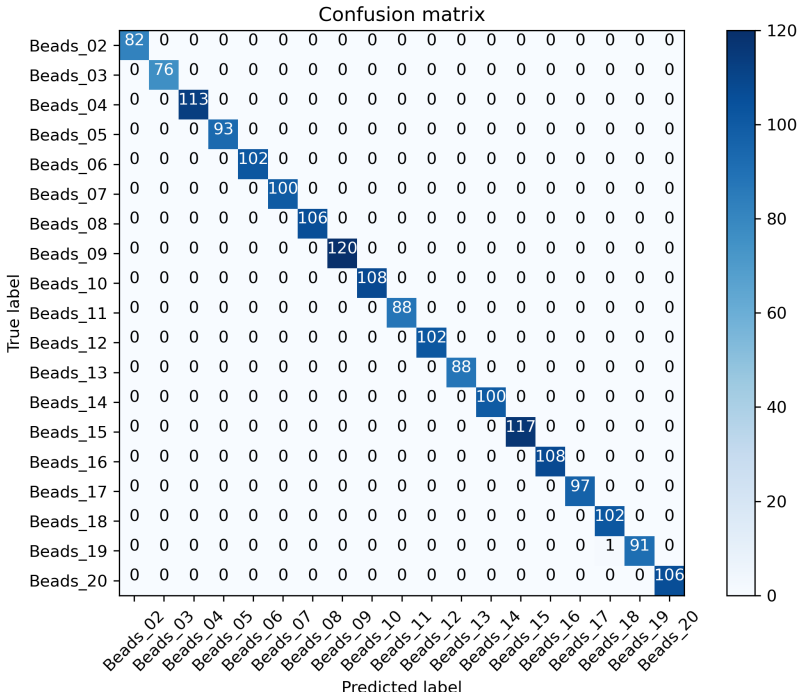

Figure 4.4.1.1. Confusion Matrix of non-polydispersity system classification results using 257x257 as input coherent X-ray image size.

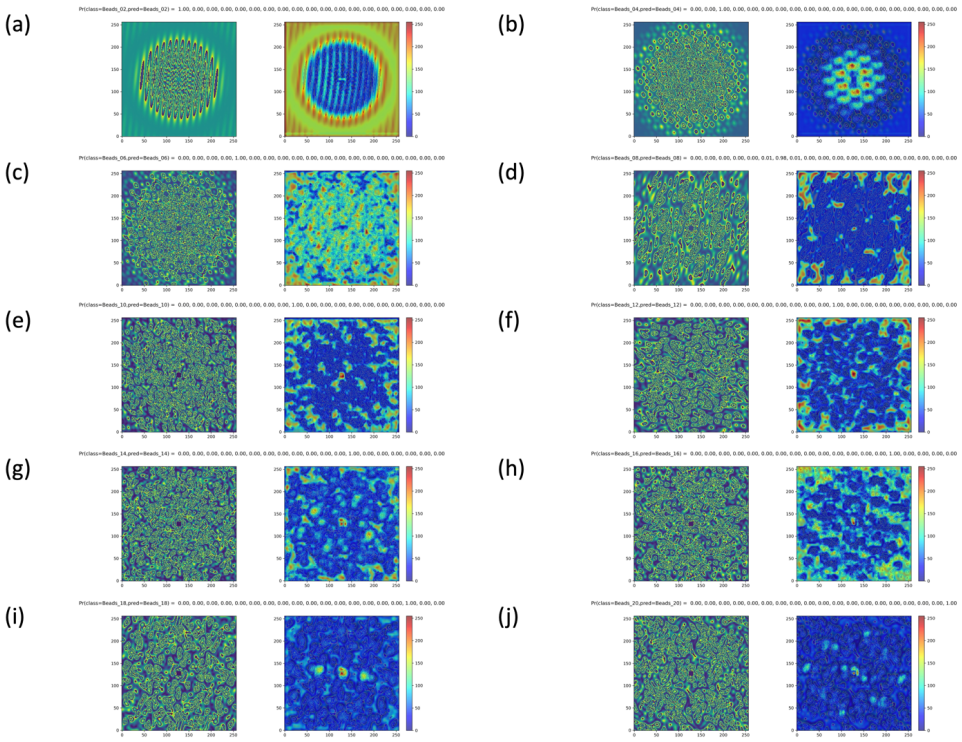

Figure 4.4.1.2. Grad Class Activation Map Examples of non-polydispersity systems where (a), (b), (c), (d), (e), (f), (g), (h), (i), (j) corresponding to beads number 2, 4, 6, 8, 10, 12, 14, 16, 18, 20.

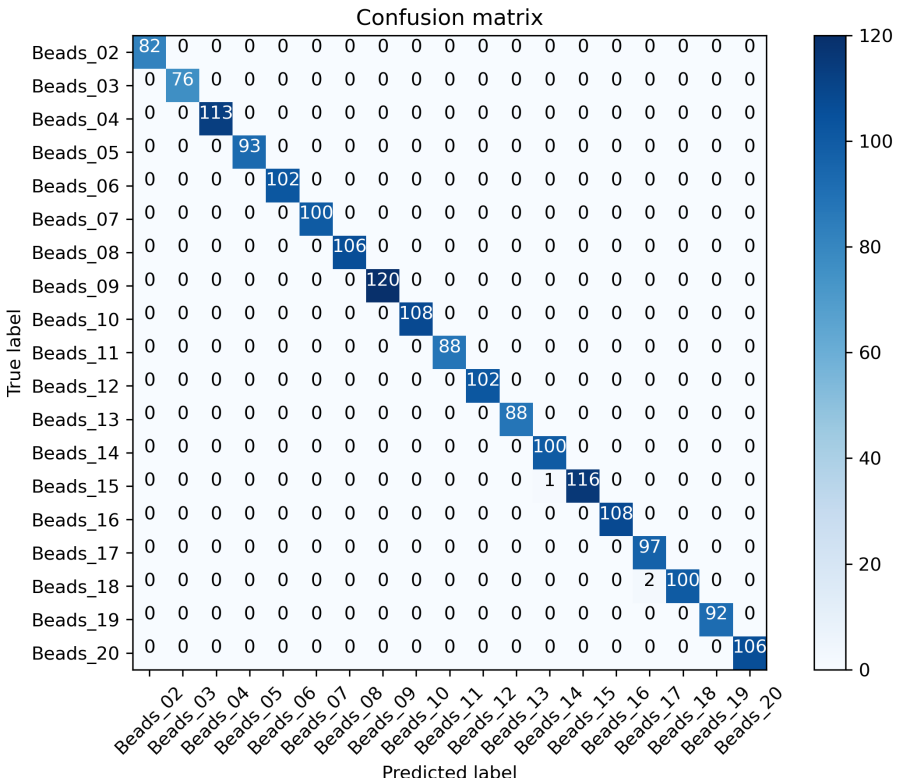

Figure 4.4.1.3. Confusion Matrix of non-polydispersity system classification results using 513x513 as input coherent X-ray image size.

4.4.2 Polydispersity System Results

In previous section, we presented results of the deep learning classifier for coherent X-ray imaging patterns in non-polydisperse systems. However, the non-disperse setting is ideal and nearly impossible to obtain in real world settings, so to further test the capacity of the classifier, we retrained the model and tested it on a polydisperse dataset, and the results are shown below. In **Figure 4.4.2.1**, we show the confusion matrix of the polydisperse classifier. We can see there is a significant performance drop compared to non-polydisperse results. To better illustrate the cases where the classifier has errors, we made sure each class has the same number, 100 in our

setting, of testing images. Then in **Table 4.4.2.1**, we showed class specific precision, recall, F1 scores. The cross-class precision, recall and F1-score are all 0.89 due to the drop of performance in large bead number systems e.g., Bead number 14 to 20.

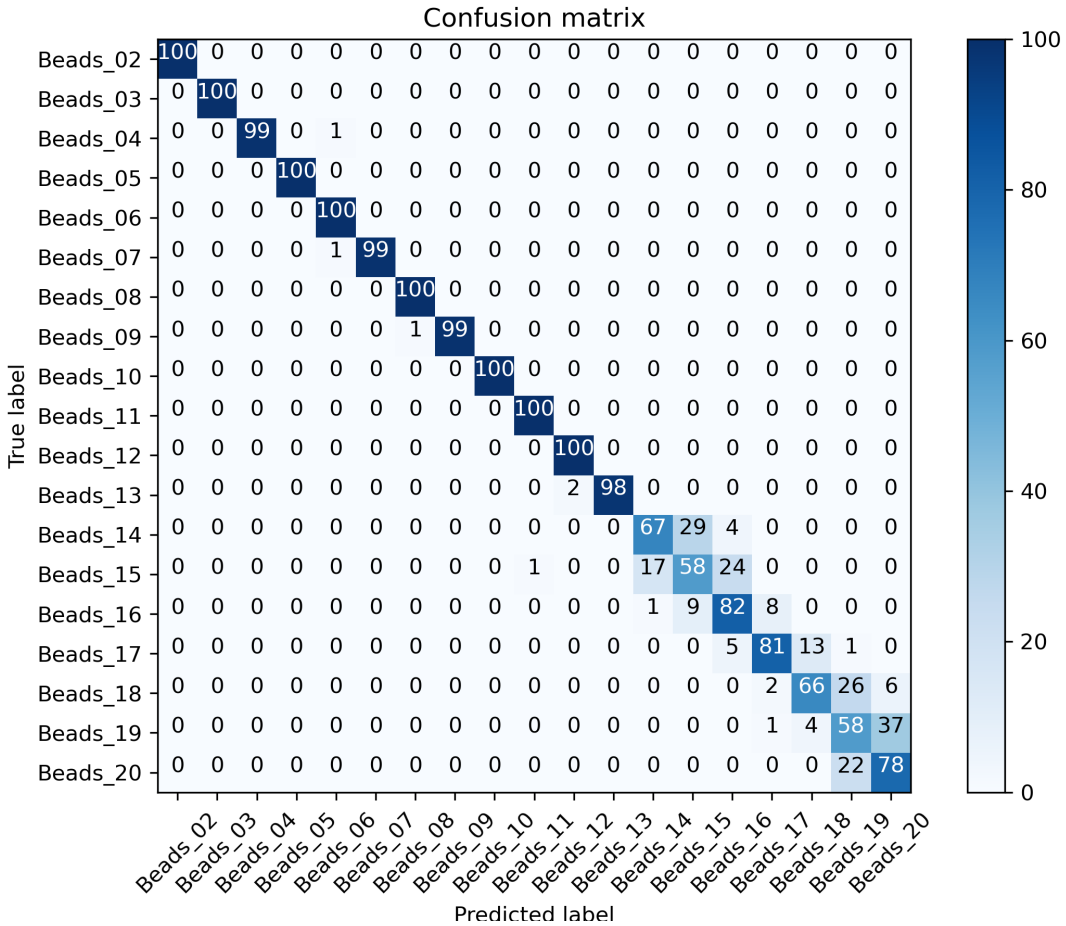

Figure 4.4.2.1. Confusion Matrix of polydispersity system classification results using 257x257 as input coherent X-ray image size.

Table 4.4.2.1. Classification performance for different polydispersity bead numbers.

Beads Numbers	precision	recall	F1-score
Beads_02	1	1	1

Beads_03	1	1	1
Beads_04	1	0.99	0.99
Beads_05	1	1	1
Beads_06	0.98	1	0.99
Beads_07	1	0.99	0.99
Beads_08	0.99	1	1
Beads_09	1	0.99	0.99
Beads_10	1	1	1
Beads_11	0.99	1	1
Beads_12	0.98	1	0.99
Beads_13	1	0.98	0.99
Beads_14	0.79	0.67	0.72
Beads_15	0.6	0.58	0.59
Beads_16	0.71	0.82	0.76
Beads_17	0.88	0.81	0.84
Beads_18	0.8	0.66	0.72
Beads_19	0.54	0.58	0.56
Beads_20	0.64	0.78	0.71

To better understand the reasoning why classifier performances dropped for larger bead numbers, we checked the Grad-CAM plots (shown in **Figure 4.4.2.2**). In **Figure 4.4.2.2**, as the beads number went up, the features in Grad-CAM would be difficult to distinguish which reflected the drop of deep learning classifier performance. One possible reason could be that

polydispersity would generate more scattering centers of different sizes and thus reduces the contrast associated with peaks in patterns of coherent X-ray imaging and hence made it difficult to learn about correct patterns for larger bead numbers[117].

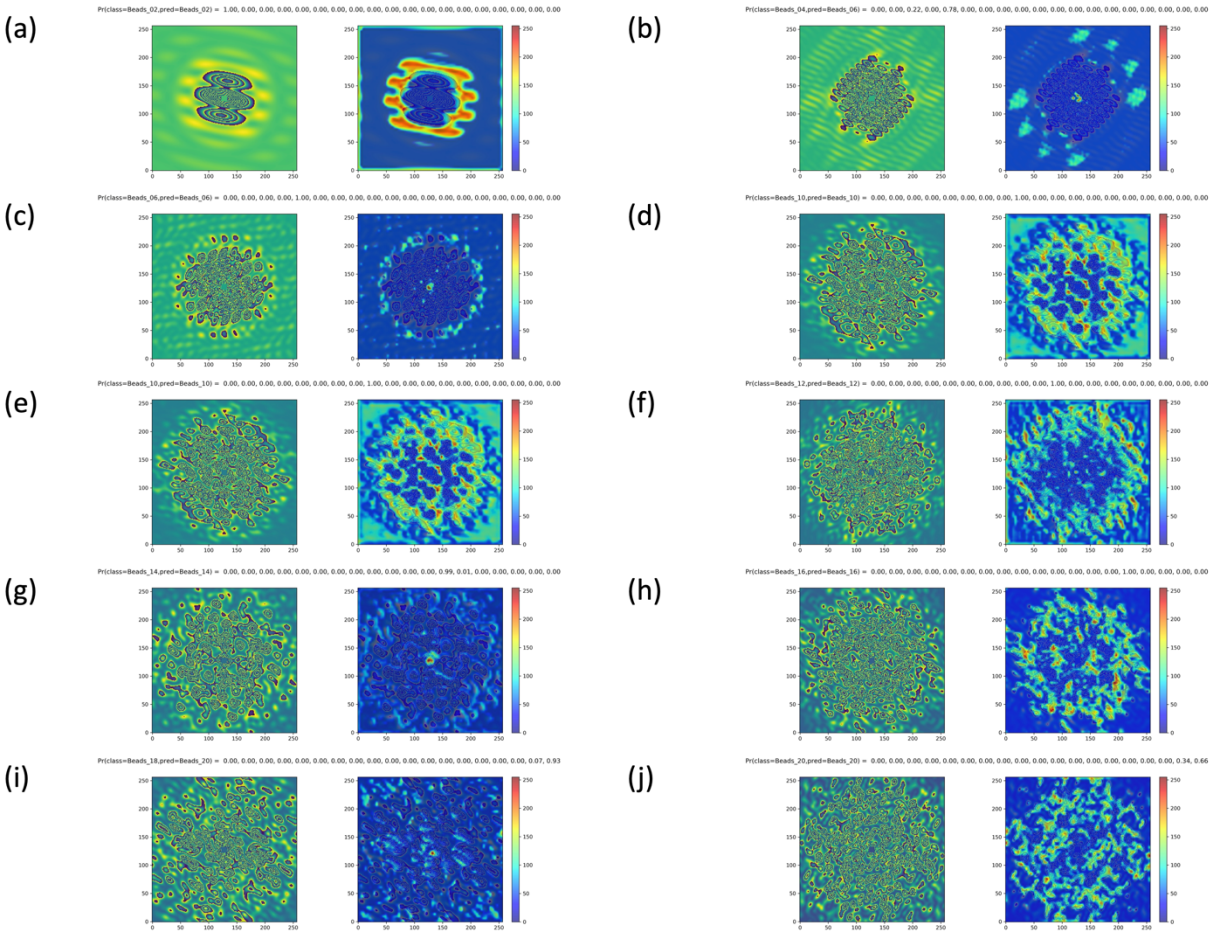

Figure 4.4.2.2. Grad Class Activation Map Examples of polydispersity systems where (a), (b), (c), (d), (e), (f), (g), (h), (i), (j) corresponding to beads number 2, 4, 6, 8, 10, 12, 14, 16, 18, 20.

4.5 Conclusions

We have developed a deep learning based model to identify disk number by classification from coherent X-ray imaging patterns of a two dimensional disk model system. The classifier was tested for cases with and without dispersity and it shown to be effective in both, although

systems with dispersity show significant errors depending on the scale of the dispersity. We also sought to understand the mechanisms of the deep learning classifier with Grad-CAM. The Grad-CAM analysis showed that certain features contained in X-ray image patterns could aid the interpretation and understanding of the classification results. Overall, our results demonstrate that without applying complex experimental procedures e.g., taking multiple images in different angles or positions for phase retrieval or pre-defined assumptions of samples, we could directly extract real space sample information like number density from k-space X-ray image patterns generated by Fourier Transforms and there are few tools to link information between real space and image spaces[150,151].

As a proof-of-concept study, our approach has some limitations that needs to be solved by future studies. First, real coherent X-ray images have noises due to both environments, devices, detectors etc. and the impacts of noise on the classification will need to be assessed, although it is likely that it does not represent a fundamental challenge. We expect that enough data with well-enough controlled noise will obtain robust models, as found for most of the data here. Second, currently, our model fails in larger bead number polydispersity system, which is at least in part since many beads with different sizes blur peaks and thus make it hard to discover patterns. This problem could potentially be solved by adding more training images for larger bead numbers. A closely related third limitation is that it is still unclear what kinds of training are needed to enable extraction of useful data e.g., how many training images are needed to train a successful pattern extraction model and can such training data be practically obtained. Finally, we note a fourth limitation and major limitation, which is that our system is a model 2D system. This approach needs to be tested on data of 3D systems and if possible, tested with real-world experimental data e.g., colloid system which is easy to track and investigate[98].

Chapter 5 : Other UW-Madison Research Summary

5.1 Chapter Abstract

In this chapter, I will summarize my research that is not the core part of this thesis but the skills I learned and used are the same and they are a part of my Ph.D. research experience. A Ph.D. study is a long journey and I have spent time across different departments and worked in multidisciplinary studies. Those are all valuable experiences and closely connected to the general machine learning research theme of my thesis. I will briefly summarize each research effort and more details could be found from the published papers.

5.2 Medical ML

I worked with Dr. Meghan Lubner of UW-Health to develop and evaluate machine learning algorithms that can help radiologists to delineate cancers using non-invasive CT-based radiomics for renal cell carcinoma (RCC)[152] and pancreatic cysts (PCs)[153]. And the published papers are listed below,

- Awe, Adam M., Michael M. Vanden Heuvel, Tianyuan Yuan, Victoria R. Rendell, **Mingren Shen**, Agrima Kampani, Shanchao Liang, Dane D. Morgan, Emily R. Winslow, and Meghan G. Lubner. "Machine learning principles applied to CT radiomics to predict mucinous pancreatic cysts." Abdominal Radiology (2021): 1-11.
- Gurbani, Sidharth, Dane Morgan, Varun Jog, Leo Dreyfuss, **Mingren Shen**, Arighno Das, E. Jason Abel, and Meghan G. Lubner. "Evaluation of radionics and machine learning in the identification of aggressive tumor features in renal cell carcinoma (RCC)." Abdominal Radiology (2021): 1-11.

I also helped develop GAN-based segmentation tools that can help segmentation problems for multi-domain MRI images and the published paper is listed below,

- Liu, Yilin, Gregory R. Kirk, Brendon M. Nacewicz, Martin A. Styner, **Mingren Shen**, Dong Nie, Nagesh Adluru, Benjamin Yeske, Peter A. Ferrazzano, and Andrew L. Alexander. "Harmonization and Targeted Feature Dropout for Generalized Segmentation: Application to Multi-site Traumatic Brain Injury Images." In Domain Adaptation and Representation Transfer and Medical Image Learning with Less Labels and Imperfect Data, pp. 81-89. Springer, Cham, 2019.

5.3 Material Informatics

I also worked with undergraduate students from Informatics Skunkworks to apply machine learning methods for predicting material properties e.g., flash point and the published paper is listed below,

- Xiaoyu Sun, Nathaniel J. Krakauer, Alexander Politowicz, WeiTing Chen, Qiying Li, Zuoyi Li, Xianjia Shao, Alfred Sunaryo, **Mingren Shen**, James Wang, Dane Morgan. "Assessing Graph based Deep Learning Models for Predicting Flash Point." Molecular Informatics (2020), 39, 1900101.

5.4 BioPhysics

Before joining Prof. Morgan's group for Ph.D., I also did laboratory rotations in Prof. Qiang Cui's lab, now a Professor at Boston University, where I researched on soft matter physics in DNA[154] and colloids[155–157] and the published papers are listed below,

- Luo, Guan-Zheng, Ziyang Hao, Liangzhi Luo, **Mingren Shen**, Daniela Sparvoli, Yuqing Zheng, Zijie Zhang et al. "N 6-methyldeoxyadenosine directs nucleosome positioning in Tetrahymena DNA." Genome biology 19, no. 1 (2018): 200.
- **Mingren Shen**, Rui Liu, Ke Chen, and Mingcheng Yang. "Diffusive-Flux-Driven Microturbines by Fore-and-Aft Asymmetric Phoresis." Physical Review Applied 12, no. 3 (2019): 034051.

5.5 Conclusions

The papers listed in this chapter might not be material science related but they provided great experiences to learn new things and apply all physical, mathematical, computational, machine learning tools I have learned. I listed them here as they are also part of my Ph.D. study and they all helped me to complete the thesis main theme in various ways.

Chapter 6 : Summary and Future Work

6.1 Summary

In this thesis, we discussed two aspects about applying deep learning models to material science studies e.g., finding the location and geometry of different defect clusters in irradiated steels and mapping k-space X-ray image patterns with real space sample information without complex iterative algorithms. We show that a deep learning based analysis system has a performance comparable to human analysis with relatively small training data sets. This study proves the promising ability to apply deep learning to assist the development of automated data analysis data and paves the way for fast, scalable, and reliable analysis systems for massive amounts of modern material science data.

The research contained Chapter 3 regarding the using deep learning models e.g., Faster R-CNN, YOLO, Mask R-CNN to analyze defects in microscopy images or videos of alloys is summarized in this section. Our work shows that automatically detecting and tracking interesting microstructures and properties contained in TEM images and videos is viable and opens new doors for evaluating materials microstructure changes and dynamics in human labeling hardly achieved accuracy and consistency.

The contained Chapter 4 regarding the using deep learning models to extract sample information from coherent X-ray imaging patterns in this section. Our work shows deep learning can build the relationship between real space sample information e.g., number density and k-space images, and visualization methods like Class Activation Maps can help people identify important regions which could be a useful way to understand the hidden information in coherent X-ray image patterns.

6.2 Future Work

As shown in this thesis, specific questions are selected to test the ability to apply deep learning tools to material science problems. While the prominent potential of such an approach is demonstrated, some future developments are needed to fully utilize this powerful approach.

Three possible directions are given here:

- **Dataset Generation**

  Currently, in the defect analysis project, manual labeling data was used to train deep learning models, however, human errors and biases will be avoided in the trained models. Furthermore, human labeling made it difficult to extend deep learning models to new types of tasks that might not have efficient resources or methods to label data by humans. Possible solutions might be using simulation or other automatic generation methods to generate datasets[158] with labels or developing methods that do not need certain labels but rely on inner structures of data e.g., deep image prior uses realistic image priors from a single image itself to guide image restoration[159].

- **Model Optimization and Serving**

  All deep learning models present in this thesis are not optimized for high-performance consideration, e.g., automatic neural network searching[160], model compression and acceleration[161], and model pruning[162]. Those optimizations could significantly improve model speed and accuracy and hence help scale up of usage. And another important

future direction would be establishing the common sharing practice of codes and models. More and more people in the material science community are involving deep learning studies which raise reproducible and portable issues between different model implementations. Model serving of a collection of easy-to-use, standardized, and high-performance deep learning models with pre-trained weights would be a tremendous help for scientists and engineers in material sciences and other areas. Common tools used by tech companies like Google Colab[163] and Kubeflow[164] based on top of Kubernetes for MLOps[165] could be a good starting point to learn from and some model serving websites like TensorFlow-Serving[166], ModelHub.AI[167] and DLHub[168] are gradually used by the scientific community now.

- **Domain Knowledge Integration**

  Once the data and models are obtained, we can always train or build some ML or DL models but how to make sure it contributes new information, knowledge, or insights for the scientific community is a different and sometimes difficult question that needs domain experts and knowledge[169]. And more specifically, could we embed or involve domain knowledge into machine learning models to accelerate learning[170] and improve accuracy[171]. Physics-informed neural networks[172] or Physics-guided neural networks[173] and other methods[174] have shown a promising way to integrate physics and material information into ML or DL models[172].

Although limited machine learning and deep learning applications are discussed in this thesis, the potential of such an approach is demonstrated. For future developments, more easy-to-

generate and easy-to-label data, more standardized ways to share models, and more efficient integration methods of ML/DL with physics information are needed.

References

1. Shen, M. *et al.* Multi defect detection and analysis of electron microscopy images with deep learning. *Comput. Mater. Sci.* **199**, 110576 (2021).
2. Shen, M. *et al.* A deep learning based automatic defect analysis framework for In-situ TEM ion irradiations. *Comput. Mater. Sci.* **197**, 110560 (2021).
3. Jacobs, R. *et al.* Performance, Successes and Limitations of Deep Learning Semantic Segmentation of Multiple Defects in Transmission Electron Micrographs. (2021).
4. Agrawal, A. & Choudhary, A. Perspective: Materials informatics and big data: Realization of the "fourth paradigm" of science in materials science. *APL Mater.* **4**, 053208 (2016).
5. Tolle, K. M., Tansley, D. S. W. & Hey, A. J. G. The fourth Paradigm: Data-intensive scientific discovery. *Proc. IEEE* **99**, 1334–1337 (2011).
6. Wang, Y. E., Wei, G.-Y. & Brooks, D. Benchmarking TPU, GPU, and CPU Platforms for Deep Learning. (2019).
7. de Pablo, J. J. *et al.* New frontiers for the materials genome initiative. *npj Comput. Mater. 2019 51* **5**, 1–23 (2019).
8. Liu, Y. *et al.* Machine learning in materials genome initiative: A review. *J. Mater. Sci. Technol.* **57**, 113–122 (2020).
9. Lecun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
10. White, A. A. Big data are shaping the future of materials science. *MRS Bull. 2013 388* **38**, 594–595 (2013).
11. Storer, T. Bridging the Chasm: A Survey of Software Engineering Practice in Scientific Programming. *ACM Comput. Surv.* **50**, (2017).
12. Rahman, M. S., Rivera, E., Khomh, F., Guéhéneuc, Y.-G. & Lehnert, B. Machine

Learning Software Engineering in Practice: An Industrial Case Study. (2019).

13. Runeson, P., Engström, E. & Storey, M.-A. The Design Science Paradigm as a Frame for Empirical Software Engineering. *Contemp. Empir. Methods Softw. Eng.* 127–147 (2020). doi:10.1007/978-3-030-32489-6_5

14. Ackroyd, K. S. *et al.* Scientific software development at a research facility. *IEEE Softw.* **25**, 44–51 (2008).

15. Serban, A., Van Der Blom, K., Hoos, H. & Visser, J. Adoption and Effects of Software Engineering Best Practices in Machine Learning. *Proc. 14th ACM / IEEE Int. Symp. Empir. Softw. Eng. Meas.* doi:10.1145/3382494

16. Wolpert, D. H. & Macready, W. G. No free lunch theorems for optimization. *IEEE Trans. Evol. Comput.* **1**, 67–82 (1997).

17. Antwarg, L., Miller, R. M., Shapira, B. & Rokach, L. Explaining Anomalies Detected by Autoencoders Using SHAP. (2019).

18. Zhao, W., Joshi, T., Nair, V. N. & Sudjianto, A. SHAP values for Explaining CNN-based Text Classification Models. (2020).

19. Wang, D. *et al.* Towards better process management in wastewater treatment plants: Process analytics based on SHAP values for tree-based machine learning methods. *J. Environ. Manage.* **301**, 113941 (2022).

20. Meng, Y., Yang, N., Qian, Z. & Zhang, G. What Makes an Online Review More Helpful: An Interpretation Framework Using XGBoost and SHAP Values. *J. Theor. Appl. Electron. Commer. Res. 2021, Vol. 16, Pages 466-490* **16**, 466–490 (2020).

21. Lord Kelvin and the End of Physics, which he Never Predicted. Available at: https://www.bbvaopenmind.com/en/science/physics/lord-kelvin-and-the-end-of-physics-

which-he-never-predicted/. (Accessed: 29th October 2021)

22. Mitchell, T. M. (Tom M. *Machine Learning*. (McGraw-Hill, 1997).

23. Jordan, M. I. & Mitchell, T. M. Machine learning: Trends, perspectives, and prospects. *Science (80-.).* **349**, 255–260 (2015).

24. Russakovsky, O. *et al.* Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **115**, 211–252 (2015).

25. Silver, D. *et al.* Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).

26. Morgan, D. & Jacobs, R. Opportunities and Challenges for Machine Learning in Materials Science. in *Annual Review of Materials Research* **50**, 71–103 (Annual Reviews Inc., 2020).

27. Chen, C., Ye, W., Zuo, Y., Zheng, C. & Ong, S. P. Graph Networks as a Universal Machine Learning Framework for Molecules and Crystals. *Chem. Mater.* **31**, 3564–3572 (2019).

28. Sun, X. *et al.* Assessing Graph-based Deep Learning Models for Predicting Flash Point. *Mol. Inform.* **39**, 1900101 (2020).

29. Carrasquilla, J. & Melko, R. G. Machine learning phases of matter. *Nat. Phys.* **13**, 431–434 (2017).

30. Park, C. & Ding, Y. Automating material image analysis for material discovery. *MRS Commun.* **9**, 545–555 (2019).

31. Maksov, A. *et al.* Deep learning analysis of defect and phase evolution during electron beam-induced transformations in WS 2. *npj Comput. Mater.* **5**, 12 (2019).

32. Li, W., Field, K. G. & Morgan, D. Automated defect analysis in electron microscopic

images. *npj Comput. Mater.* **4**, 1–9 (2018).

33. Roberts, G. *et al.* Deep Learning for Semantic Segmentation of Defects in Advanced STEM Images of Steels. *Sci. Rep.* **9**, (2019).
34. Maksov, A. *et al.* Deep learning analysis of defect and phase evolution during electron beam-induced transformations in WS2. *npj Comput. Mater.* **5**, 12 (2019).
35. Ziatdinov, M. *et al.* Deep Learning of Atomically Resolved Scanning Transmission Electron Microscopy Images: Chemical Identification and Tracking Local Transformations. *ACS Nano* **11**, 12742–12752 (2017).
36. Holm, E. A. *et al.* Overview: Computer Vision and Machine Learning for Microstructural Characterization and Analysis. *Metall. Mater. Trans. A 2020 5112* **51**, 5985–5999 (2020).
37. Liu, L. *et al.* Deep Learning for Generic Object Detection: A Survey. *Int. J. Comput. Vis.* (2019). doi:10.1007/s11263-019-01247-4
38. Szeliski, R. *Computer Vision: Algorithms and Applications*. (Springer London, 2011). doi:10.1007/978-1-84882-935-0
39. Forsyth, D. A. & Ponce, J. *Computer Vision: A Modern Approach*. (Pearson, 2003).
40. Zhao, Z.-Q., Zheng, P., Xu, S.-T. & Wu, X. Object Detection With Deep Learning: A Review. *IEEE Trans. Neural Networks Learn. Syst.* 1–21 (2019). doi:10.1109/TNNLS.2018.2876865
41. Zou, Z., Shi, Z., Guo, Y. & Ye, J. Object Detection in 20 Years: A Survey. 1–39 (2019).
42. Ren, S., He, K., Girshick, R. & Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **39**, 1137–1149 (2015).
43. Redmon, J., Divvala, S., Girshick, R. & Farhadi, A. You Only Look Once: Unified, Real-

Time Object Detection. (2015).

44. Oktay, O. *et al.* Attention U-Net: Learning Where to Look for the Pancreas. (2018).

45. Çiçek, Ö., Abdulkadir, A., Lienkamp, S. S., Brox, T. & Ronneberger, O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **9901 LNCS**, 424–432 (2016).

46. Ronneberger, O., Fischer, P. & Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **9351**, 234–241 (2015).

47. Falk, T. *et al.* U-Net: deep learning for cell counting, detection, and morphometry. *Nat. Methods 2018 161* **16**, 67–70 (2018).

48. Zhou, Z., Rahman Siddiquee, M. M., Tajbakhsh, N. & Liang, J. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **11045 LNCS**, 3–11 (2018).

49. Du, G., Cao, X., Liang, J., Chen, X. & Zhan, Y. Medical image segmentation based on U-Net: A review. *J. Imaging Sci. Technol.* **64**, (2020).

50. Carion, N. *et al.* End-to-End Object Detection with Transformers. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **12346 LNCS**, 213–229 (2020).

51. Zhu, X. *et al.* Deformable DETR: Deformable Transformers for End-to-End Object Detection. (2020).

52. Guan, T. *et al.* M3DeTR: Multi-representation, Multi-scale, Mutual-relation 3D Object Detection with Transformers. (2021).

53. Field, K. G., Hu, X., Littrell, K. C., Yamamoto, Y. & Snead, L. L. Radiation tolerance of neutron-irradiated model Fe-Cr-Al alloys. *J. Nucl. Mater.* **465**, 746–755 (2015).

54. Zinkle, S. J. & Busby, J. T. Structural materials for fission & fusion energy. *Materials Today* **12**, 12–19 (2009).

55. Jenkins, M. . & Kirk, M. . *Characterisation of Radiation Damage by Transmission Electron Microscopy*. *Iop* **20002352**, (Taylor & Francis, 2000).

56. Yao, B., Edwards, D. J. & Kurtz, R. J. TEM characterization of dislocation loops in irradiated bcc Fe-based steels. *J. Nucl. Mater.* **434**, 402–410 (2013).

57. Liu, L. *et al.* Deep Learning for Generic Object Detection: A Survey. (2018).

58. Field, K. G. *et al.* Heterogeneous dislocation loop formation near grain boundaries in a neutron-irradiated commercial FeCrAl alloy. *J. Nucl. Mater.* **483**, 54–61 (2017).

59. Field, K. G., Briggs, S. A., Sridharan, K., Yamamoto, Y. & Howard, R. H. Dislocation loop formation in model FeCrAl alloys after neutron irradiation below 1 dpa. *J. Nucl. Mater.* **495**, 20–26 (2017).

60. Schindelin, J. *et al.* Fiji: An open-source platform for biological-image analysis. *Nat. Methods* **9**, 676–682 (2012).

61. Schneider, C. A., Rasband, W. S. & Eliceiri, K. W. NIH Image to ImageJ: 25 years of image analysis. *Nat. Methods* **9**, 671–675 (2012).

62. Niitani, Y., Ogawa, T., Saito, S. & Saito, M. ChainerCV: a Library for Deep Learning in Computer Vision. 1217–1220 (2017). doi:10.1145/3123266.3129395

63. Module: exposure — skimage v0.18.0 docs. Available at: https://scikit-image.org/docs/stable/api/skimage.exposure.html#skimage.exposure.equalize_adapthist. (Accessed: 23rd April 2021)

64. Module: filters — skimage v0.19.0.dev0 docs. Available at: https://scikit-image.org/docs/dev/api/skimage.filters.html#skimage.filters.gaussian. (Accessed: 23rd April 2021)

65. Shorten, C. & Khoshgoftaar, T. M. A survey on Image Data Augmentation for Deep Learning. *J. Big Data* **6**, 60 (2019).

66. Garrido, G. & Joshi, P. *OpenCV 3.x with Python By Example Second Edition Make the most of OpenCV and Python to build applications for object recognition and augmented reality*. (2018).

67. Pan, S. J. & Yang, Q. A Survey on Transfer Learning. *IEEE Trans. Knowl. Data Eng.* **22**, 1345–1359 (2010).

68. Kornilov, A. & Safonov, I. An Overview of Watershed Algorithm Implementations in Open Source Libraries. *J. Imaging* **4**, 123 (2018).

69. OpenCV: Image Segmentation with Watershed Algorithm. Available at: https://docs.opencv.org/master/d3/db4/tutorial_py_watershed.html. (Accessed: 24th March 2021)

70. opencv/fitellipse.py at master · kipr/opencv. Available at: https://github.com/kipr/opencv/blob/master/samples/python/fitellipse.py. (Accessed: 21st April 2021)

71. Jesse, S. *et al.* Big Data Analytics for Scanning Transmission Electron Microscopy Ptychography. *Sci. Rep.* **6**, 26348 (2016).

72. Dean, J. *et al.* Large Scale Distributed Deep Networks. in *Advances in Neural Information Processing Systems 25* (eds. Pereira, F., Burges, C. J. C., Bottou, L. & Weinberger, K. Q.) 1223–1231 (Curran Associates, Inc., 2012).

73. Jia, Y. *et al.* Caffe. in *Proceedings of the ACM International Conference on Multimedia - MM '14* 675–678 (ACM Press, 2014). doi:10.1145/2647868.2654889

74. Cheng, Y., Wang, D., Zhou, P. & Zhang, T. A Survey of Model Compression and Acceleration for Deep Neural Networks. (2017).

75. Chen, P.-H. C. *et al.* An augmented reality microscope with real-time artificial intelligence integration for cancer diagnosis. *Nat. Med.* 1–5 (2019). doi:10.1038/s41591-019-0539-7

76. Taheri, M. L. *et al.* Current status and future directions for in situ transmission electron microscopy. *Ultramicroscopy* **170**, 86–95 (2016).

77. Sinclair, R. In-situ high-resolution transmission electron microscopy of material reactions. *MRS Bulletin* **38**, 1065–1071 (2013).

78. DeCost, B. L., Jain, H., Rollett, A. D. & Holm, E. A. Computer Vision and Machine Learning for Autonomous Characterization of AM Powder Feedstocks. *Jom* **69**, 456–465 (2017).

79. Gola, J. *et al.* Advanced microstructure classification by data mining methods. *Comput. Mater. Sci.* **148**, 324–335 (2018).

80. Groom, D. J. *et al.* Automatic segmentation of inorganic nanoparticles in BF TEM micrographs. *Ultramicroscopy* **194**, 25–34 (2018).

81. DeCost, B. L., Francis, T. & Holm, E. A. Exploring the microstructure manifold: Image texture representations applied to ultrahigh carbon steel microstructures. *Acta Mater.* **133**, 30–40 (2017).

82. DeCost, B. L. & Holm, E. A. A large dataset of synthetic SEM images of powder materials and their ground truth 3D structures. *Data Br.* **9**, 727–731 (2016).

83. de la Rosa-Trevín, J. M. *et al.* Scipion: A software framework toward integration, reproducibility and validation in 3D electron microscopy. *J. Struct. Biol.* **195**, 93–99 (2016).

84. Ramírez-Aportela, E. *et al.* Automatic local resolution-based sharpening of cryo-EM maps. *Bioinformatics* (2019). doi:10.1093/bioinformatics/btz671

85. Ramírez-Aportela, E., Mota, J., Conesa, P., Carazo, J. M. & Sorzano, C. O. S. DeepRes: A new deep-learning- and aspect-based local resolution method for electron-microscopy maps. *IUCrJ* **6**, 1054–1063 (2019).

86. Tseng, Q. *et al.* A new micropatterning method of soft substrates reveals that different tumorigenic signals can promote or reduce cell contraction levels. *Lab Chip* **11**, 2231–2240 (2011).

87. Haley, J. C. *et al.* Dislocation loop evolution during in-situ ion irradiation of model FeCrAl alloys. *Acta Mater.* **136**, 390–401 (2017).

88. Parish, C. M., Field, K. G., Certain, A. G. & Wharry, J. P. Application of STEM characterization for investigating radiation effects in BCC Fe-based alloys. *Journal of Materials Research* **30**, 1275–1289 (2015).

89. Tseng, Q. *et al.* Spatial organization of the extracellular matrix regulates cell-cell junction positioning. *Proc. Natl. Acad. Sci. U. S. A.* **109**, 1506–1511 (2012).

90. Qingzong Tseng. Template Matching and Slice Alignment-- ImageJ Plugins. (2015).

91. Zhao, L. & Li, S. Object Detection Algorithm Based on Improved YOLOv3. *Electronics* **9**, 537 (2020).

92. Van Etten, A. You Only Look Twice: Rapid Multi-Scale Object Detection In Satellite Imagery. *arXiv* (2018).

93. Bergner, F., Pareige, C., Hernández-Mayoral, M., Malerba, L. & Heintze, C. Application of a three-feature dispersed-barrier hardening model to neutron-irradiated Fe-Cr model alloys. *J. Nucl. Mater.* **448**, 96–102 (2014).

94. Was, G. S. *Fundamentals of radiation materials science: Metals and alloys*. *Fundamentals of Radiation Materials Science: Metals and Alloys* (Springer Berlin Heidelberg, 2007). doi:10.1007/978-3-540-49472-0

95. Bradski, G. The OpenCV Library. *Dr. Dobb's J. Softw. Tools* (2000).

96. Li, P., Wang, D., Wang, L. & Lu, H. Deep visual tracking: Review and experimental comparison. *Pattern Recognit.* **76**, 323–338 (2018).

97. Ali, A. *et al.* Visual object tracking—classical and contemporary approaches. *Frontiers of Computer Science* **10**, 167–188 (2016).

98. Crocker, J. C. & Grier, D. G. Methods of Digital Video Microscopy for Colloidal Studies. *J. Colloid Interface Sci.* **179**, 298–310 (1996).

99. Allan, D., Caswell, T., Keim, N. & van der Wel, C. trackpy: Trackpy v0.3.2. (2016). doi:10.5281/ZENODO.60550

100. Arakawa, K. *et al.* Observation of the one-dimensional diffusion of nanometer-sized dislocation loops. *Science (80-.).* **318**, 956–959 (2007).

101. Du, Z., Yin, J. & Yang, J. Expanding Receptive Field YOLO for Small Object Detection. in *Journal of Physics: Conference Series* **1314**, 012202 (Institute of Physics Publishing, 2019).

102. Redmon, J. & Farhadi, A. YOLO9000: Better, faster, stronger. in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017* **2017-January**, 6517–6525 (Institute of Electrical and Electronics Engineers Inc., 2017).

103. Kirkland, E. J. *Advanced Computing in Electron Microscopy*. (2010).

104. Goodfellow, I. NIPS 2016 Tutorial: Generative Adversarial Networks. (2016).

105. Pan, Z. *et al.* Recent Progress on Generative Adversarial Networks (GANs): A Survey. *IEEE Access* **7**, 36322–36333 (2019).

106. Huang, S. W. *et al.* AugGAN: Cross domain adaptation with GAN-based data augmentation. in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **11213 LNCS**, (2018).

107. Liu, Y. *et al.* Harmonization and targeted feature dropout for generalized segmentation: Application to multi-site traumatic brain injury images. in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **11795 LNCS**, 81–89 (Springer, 2019).

108. Lawrence, N., Shen, M., Yin, R., Feng, C. & Morgan, D. Exploring Generative Adversarial Networks for Image-to-Image Translation in STEM Simulation. (2020).

109. Blaiszik, B. *et al.* A data ecosystem to support machine learning in materials science. *MRS Commun.* **9**, 1125–1133 (2019).

110. Blaiszik, B. *et al.* The Materials Data Facility: Data Services to Advance Materials Science Research. *JOM* **68**, 2045–2052 (2016).

111. He, K., Gkioxari, G., Dollar, P. & Girshick, R. Mask R-CNN. *Proc. IEEE Int. Conf. Comput. Vis.* **2017-Octob**, 2980–2988 (2017).

112. Wu,Yuxin;Kirillov,Alexander;Massa,Francisco;Lo,Wan-Yen;Girshick, R. Detectron2. (2019). Available at: https://github.com/facebookresearch/detectron2. (Accessed: 14th October 2021)

113. Field, K. G. *et al.* Development and Deployment of Automated Machine Learning

Detection in Electron Microcopy Experiments. *Microsc. Microanal.* **27**, 2136–2137 (2021).

114. Brisard, S., Serdar, M. & Monteiro, P. J. M. Multiscale X-ray tomography of cementitious materials: A review. *Cem. Concr. Res.* **128**, 105824 (2020).

115. Leitner, M., Sepiol, B., Stadler, L.-M., Pfau, B. & Vogl, G. Atomic diffusion studied with coherent X-rays. *Nat. Mater. 2009 89* **8**, 717–720 (2009).

116. Madsen, A., Fluerasu, A. & Ruta, B. Structural Dynamics of Materials Probed by X-Ray Photon Correlation Spectroscopy. *Synchrotron Light Sources Free. Lasers Accel. Physics, Instrum. Sci. Appl.* 1617–1641 (2016). doi:10.1007/978-3-319-14394-1_29

117. Boldon, L., Laliberte, F. & Liu, L. Review of the fundamental theories behind small angle X-ray scattering, molecular dynamics simulations, and relevant integrated application. *http://dx.doi.org/10.3402/nano.v6.25661* **6**, 25661 (2015).

118. Sandy, A. R., Zhang, Q. & Lurio, L. B. Hard X-Ray Photon Correlation Spectroscopy Methods for Materials Studies. *https://doi.org/10.1146/annurev-matsci-070317-124334* **48**, 167–190 (2018).

119. Sheyfer, D. *et al.* Nanoscale Critical Phenomena in a Complex Fluid Studied by X-Ray Photon Correlation Spectroscopy. *Phys. Rev. Lett.* **125**, 125504 (2020).

120. Evenson, Z. *et al.* X-Ray Photon Correlation Spectroscopy Reveals Intermittent Aging Dynamics in a Metallic Glass. *Phys. Rev. Lett.* **115**, 175701 (2015).

121. Hoheisel, M. Review of medical imaging with emphasis on X-ray detectors. *Nucl. Instruments Methods Phys. Res. Sect. A Accel. Spectrometers, Detect. Assoc. Equip.* **563**, 215–224 (2006).

122. Cnudde, V. & Boone, M. N. High-resolution X-ray computed tomography in geosciences:

A review of the current technology and applications. *Earth-Science Rev.* **123**, 1–17 (2013).

123. Miao, J., Ishikawa, T., Robinson, I. K. & Murnane, M. M. Beyond crystallography: Diffractive imaging using coherent X-ray light sources. *Science (80-.).* **348**, 530–535 (2015).

124. Chapman, H. N. X-ray imaging beyond the limits. *Nat. Mater. 2009 84* **8**, 299–301 (2009).

125. Jiang, H. *et al.* Three-Dimensional Coherent X-Ray Diffraction Imaging of Molten Iron in Mantle Olivine at Nanoscale Resolution. *Phys. Rev. Lett.* **110**, 205501 (2013).

126. Clark, J. N. *et al.* Ultrafast Three-Dimensional Imaging of Lattice Dynamics in Individual Gold Nanocrystals. *Science (80-.).* **341**, 56–59 (2013).

127. Jiang, H. *et al.* Quantitative 3D imaging of whole, unstained cells by using X-ray diffraction microscopy. *Proc. Natl. Acad. Sci.* **107**, 11234–11239 (2010).

128. Ekeberg, T. *et al.* Three-Dimensional Reconstruction of the Giant Mimivirus Particle with an X-Ray Free-Electron Laser. *Phys. Rev. Lett.* **114**, 098102 (2015).

129. Lohse, L. M. *et al.* A phase-retrieval toolbox for X-ray holography and tomography. *urn:issn:1600-5775* **27**, 852–859 (2020).

130. Maddali, S. *et al.* Phase retrieval for Bragg coherent diffraction imaging at high x-ray energies. *Phys. Rev. A* **99**, 053838 (2019).

131. Hagemann, J., Töpperwien, M. & Salditt, T. Phase retrieval for near-field X-ray imaging beyond linearisation or compact support. *Appl. Phys. Lett.* **113**, 041109 (2018).

132. Fannjiang, A. & Strohmer, T. The numerics of phase retrieval. *Acta Numer.* **29**, 125–228 (2020).

133. He, K., Zhang, X., Ren, S. & Sun, J. Identity mappings in deep residual networks. in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **9908 LNCS**, 630–645 (Springer Verlag, 2016).

134. He, K., Zhang, X., Ren, S. & Sun, J. Deep residual learning for image recognition. in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* **2016-Decem**, 770–778 (IEEE Computer Society, 2016).

135. Shrestha, A. & Mahmood, A. Review of Deep Learning Algorithms and Architectures. *IEEE Access* **7**, 53040–53065 (2019).

136. Chollet, F. *Deep Learning with Python*. *Shelter Island, NY : Manning Publications Co., [2018] ©2018* (Manning Publications Co, 2018).

137. Abadi, M. *et al.* TensorFlow: A System for Large-Scale Machine Learning. in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)* 265–284 (2016). doi:10.1038/nn.3331

138. Selvaraju, R. R. *et al.* Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *Proc. IEEE Int. Conf. Comput. Vis.* **2017-October**, 618–626 (2017).

139. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A. & Torralba, A. Learning Deep Features for Discriminative Localization. *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.* **2016-December**, 2921–2929 (2016).

140. Li, Y., Yang, H., Li, J., Chen, D. & Du, M. EEG-based intention recognition with deep recurrent-convolution neural network: Performance and channel selection by Grad-CAM. *Neurocomputing* **415**, 225–233 (2020).

141. He, T. *et al.* MediMLP: Using Grad-CAM to Extract Crucial Variables for Lung Cancer

Postoperative Complication Prediction. *IEEE J. Biomed. Heal. Informatics* **24**, 1762–1771 (2020).

142. Jiang, H. *et al.* A Multi-Label Deep Learning Model with Interpretable Grad-CAM for Diabetic Retinopathy Classification. *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS* **2020-July**, 1560–1563 (2020).
143. Zhang, Y. *et al.* Grad-CAM helps interpret the deep learning models trained to classify multiple sclerosis types using clinical brain magnetic resonance imaging. *J. Neurosci. Methods* **353**, 109098 (2021).
144. Panwar, H. *et al.* A deep learning and grad-CAM based color visualization approach for fast detection of COVID-19 cases using chest X-ray and CT-Scan images. *Chaos, Solitons & Fractals* **140**, 110190 (2020).
145. Kim, J., Oh, J. & Heo, T. Y. Acoustic Scene Classification and Visualization of Beehive Sounds Using Machine Learning Algorithms and Grad-CAM. *Math. Probl. Eng.* **2021**, (2021).
146. Joo, H. T. & Kim, K. J. Visualization of deep reinforcement learning using Grad-CAM: How AI plays atari games? *IEEE Conf. Comput. Intell. Games, CIG* **2019-August**, (2019).
147. Borg, M. *et al.* Test automation with grad-CAM Heatmaps - A future pipe segment in MLOps for Vision AI? *Proc. - 2021 IEEE 14th Int. Conf. Softw. Testing, Verif. Valid. Work. ICSTW 2021* 175–181 (2021). doi:10.1109/ICSTW52544.2021.00039
148. Truong, L. P., Pham, B. D. & Vu, Q. H. A Mobile Deep Convolutional Neural Network Combined with Grad-CAM Visual Explanations for Real Time Tomato Quality Classification System. *Proc. 2020 5th Int. Conf. Green Technol. Sustain. Dev. GTSD 2020* 321–325 (2020). doi:10.1109/GTSD50082.2020.9303079

149. keras-vis/class_activation_maps.md at 90ae5565951b5e6a90d706b8205c2c4dfc271505 · raghakot/keras-vis. Available at: https://github.com/raghakot/keras-vis/blob/90ae5565951b5e6a90d706b8205c2c4dfc271505/docs/templates/visualizations/class_activation_maps.md. (Accessed: 18th August 2021)

150. Jiang, W., Zhang, Y. & Dai, Q. Parameterized reconstruction based Fourier Ptychography. *Proc. - IEEE Int. Conf. Multimed. Expo* **2016-August**, (2016).

151. Zhang, Y. *et al.* Self-learning based Fourier ptychographic microscopy. *Opt. Express, Vol. 23, Issue 14, pp. 18471-18486* **23**, 18471–18486 (2015).

152. Gurbani, S. *et al.* Evaluation of radiomics and machine learning in identification of aggressive tumor features in renal cell carcinoma (RCC). *Abdom. Radiol.* **46**, 4278–4288 (2021).

153. Awe, A. M. *et al.* Machine learning principles applied to CT radiomics to predict mucinous pancreatic cysts. *Abdom. Radiol.* **1**, 1–11 (2021).

154. Luo, G. Z. *et al.* N 6 -methyldeoxyadenosine directs nucleosome positioning in Tetrahymena DNA. *Genome Biol.* **19**, 200 (2018).

155. Shen, M., Liu, R., Chen, K. & Yang, M. Diffusive-Flux-Driven Microturbines by Fore-and-Aft Asymmetric Phoresis. *Phys. Rev. Appl.* **12**, 034051 (2019).

156. Shen, M. *et al.* Chemically driven fluid transport in long microchannels. *J. Chem. Phys.* **145**, 124119 (2016).

157. Shen, M.-R., Liu, R., Hou, M.-Y., Yang, M.-C. & Chen, K. Mesoscale simulation of self-diffusiophoretic microrotor. *Wuli Xuebao/Acta Phys. Sin.* **65**, (2016).

158. Cohn, R. & Holm, E. Neural message passing for predicting abnormal grain growth in

Monte Carlo simulations of microstructural evolution. (2021).

159. Ulyanov, D., Vedaldi, A. & Lempitsky, V. Deep Image Prior. *Int. J. Comput. Vis.* **128**, 1867–1888 (2017).

160. He, X., Zhao, K. & Chu, X. AutoML: A survey of the state-of-the-art. *Knowledge-Based Syst.* **212**, 106622 (2021).

161. He, Y. *et al.* AMC: AutoML for Model Compression and Acceleration on Mobile Devices. *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)* **11211 LNCS**, 815–832 (2018).

162. He, Y., Zhang, X. & Sun, J. Channel Pruning for Accelerating Very Deep Neural Networks. *Proc. IEEE Int. Conf. Comput. Vis.* **2017-October**, 1398–1406 (2017).

163. Overview of Colaboratory Features - Colaboratory. Available at: https://colab.research.google.com/notebooks/basic_features_overview.ipynb. (Accessed: 25th October 2021)

164. Bisong, E. Deploying an End-to-End Machine Learning Solution on Kubeflow Pipelines. *Build. Mach. Learn. Deep Learn. Model. Google Cloud Platf.* 687–695 (2019). doi:10.1007/978-1-4842-4470-8_47

165. Zhou, Y., Yu, Y. & Ding, B. Towards MLOps: A Case Study of ML Pipeline Platform. *Proc. - 2020 Int. Conf. Artif. Intell. Comput. Eng. ICAICE 2020* 494–500 (2020). doi:10.1109/ICAICE51518.2020.00102

166. Olston, C. *et al.* TensorFlow-Serving: Flexible, High-Performance ML Serving. (2017).

167. Hosny, A. *et al.* ModelHub.AI: Dissemination Platform for Deep Learning Models. 15 (2019).

168. Chard, R. *et al.* DLHub: Model and data serving for science. *Proc. - 2019 IEEE 33rd Int.*

Parallel Distrib. Process. Symp. IPDPS 2019 283–292 (2019). doi:10.1109/IPDPS.2019.00038

169. Roscher, R., Bohn, B., Duarte, M. F. & Garcke, J. Explainable Machine Learning for Scientific Insights and Discoveries. *IEEE Access* **8**, 42200–42216 (2020).

170. Jiang, W. *et al.* A Two-Stage Framework for Compound Figure Separation. *Microsc. Microanal.* **27**, 1204–1208 (2021).

171. Murdock, R. J., Kauwe, S. K., Wang, A. Y.-T. & Sparks, T. D. Is Domain Knowledge Necessary for Machine Learning Materials Properties? *Integr. Mater. Manuf. Innov. 2020 93* **9**, 221–227 (2020).

172. Kadeethum, T., Jørgensen, T. M. & Nick, H. M. Physics-informed neural networks for solving nonlinear diffusivity and Biot's equations. *PLoS One* **15**, e0232683 (2020).

173. Daw, A., Karpatne, A., Watkins, W., Read, J. & Kumar, V. Physics-guided Neural Networks (PGNN): An Application in Lake Temperature Modeling. (2017).

174. Childs, C. M. & Washburn, N. R. Embedding domain knowledge for machine learning of complex material systems. *MRS Commun.* **9**, 806–820 (2019).