

LAWRENCE
LIVERMORE
NATIONAL
LABORATORY

LLNL-JRNL-859224

Phase Identification of SXRD Ti-6Al-4V using Computer Vision and Deep Learning

W. Yue, P. K. Tripahi, G. Ponon, Z. Ualikhankyzy, D. W. Brown, B. Clausen, M. Strantza, D. C. Paggan, M. A. Willard, F. Ernst, E. Ayday, V. Chaudhary, R. H. French

January 17, 2024

Integrating Materials and Manufacturing Innovation

Disclaimer

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

Phase Identification of SXRD Ti–6Al–4V using Computer Vision and Deep Learning

Weiqi Yue[1,3*], Pawan K. Tripathi[2,3], Gabriel Ponon[2,3], Zhuldyz Ualikhankyzy[1,3], Donald W. Brown[4], Bjorn Clausen[4], Maria Strantza[5], Darren C. Pagan[6], Matthew A. Willard[2,3], Frank Ernst[2,3], Erman Ayday[1,3], Vipin Chaudhary[1,3], Roger H. French[1,2,3]

[1*]Department of Computer and Data Science, Case Western Reserve University, Cleveland, 44106, OH, US.
[2]Department of Materials Science and Engineering, Case Western Reserve University, Cleveland, 44106, OH, US.
[3]Materials Data Science for Stockpile Stewardship: Center of Excellence, Case Western Reserve University, Cleveland, 44106, OH, US.
[4]Los Alamos National Laboratory, Los Alamos, 87545, NM, US.
[5]Lawrence Livermore National Laboratory, Livermore, 94550, CA, US.
[6]Department of Materials Science and Engineering, Pennsylvania State University, State College, 16801, PA, US.

*Corresponding author(s). E-mail(s): wxy215@case.edu;
Contributing authors: pkt19@case.edu; gabriel.ponon@case.edu; zxu4@case.edu; dbrown@lanl.gov; clausen@lanl.gov; strantza1@llnl.gov; dcp5303@psu.edu; maw169@case.edu; fxe5@case.edu; exa208@case.edu; vxc204@case.edu; rxf131@case.edu;

Abstract

While X-ray diffraction patterns contain information about the atomistic structure and microstructure (defect population) of materials, extracting detailed information from diffraction patterns is complex and demanding and needs to rely on prior knowledge. We hypothesize that deep-learning techniques can help to perform an effective and accurate analysis with high throughput rates. To demonstrate this concept, we applied it to determine the evolution of the β-phase volume fraction in a Ti–6Al–4V alloy during heat-treatment from video sequences

of 2D diffraction patterns recorded in transmission and with highly monochromatic radiation in a synchrotron beamline. Especially, we studied the impact of *network design* on prediction reliability and computational performance. Networks of different architectures were trained using 3012 experimental 2D patterns. A well-tuned model was found to reproduce the phase fractions of another experimental data set, consisting of 1102 diffraction patterns, with a mean-square error as small as 2.6×10^{-4}. The average prediction error of β-phase fraction was within 1.6×10^{-2} (in each diffraction pattern) of the values obtained by conventional methods. Our work demonstrates that convolutional neural networks can evaluate X-ray diffraction patterns with a remarkable level of reliability. Further, it demonstrates the significance of network design on the reliability of predictions and computational performance. The most complex models do not necessarily result in highest accuracy and may even fail to learn from the data.

Keywords: XRD, CNN, deep learning, Ti–6Al–4V, phase identification

1 Introduction

When a highly coherent (monochromatic and uni-directional) plane wave or "beam" of X-rays interacts with a solid, forced oscillation of electrons in the solid causes the emission of secondary waves. In crystallites, the periodic atomistic structure can cause the secondary waves to superimpose constructively in specific directions to produce diffracted plane waves or "reflected beams." The condition for this to occur is known as the Bragg condition:

$$\lambda = 2d_{hkl}\sin[\theta]\,, \tag{1}$$

where d_{hkl} is the spacing of the $\{hkl\}$ planes and θ is the angle they make with the primary wave vector. At distances large compared to the X-ray wavelength λ, the intensity distribution, known as a Fraunhofer diffraction pattern, shows each diffracted beam as a sharp spot. The spot pattern of a specific crystallite depends on its crystal structure and its orientation relative to the primary wave. The microstructure, i. e. the population of structural crystal defects, leads to diffuse intensity around the spots. Correspondigly, X-ray diffraction patterns can provide abundant information about crystal structures and defect populations in solids. In particular, this includes volume fractions of known crystalline phases.

A particularly promising technique for innovative materials research is 2D XRD. While 1D X-ray diffraction scans show average information from all irradiated crystallites – or grains – with suitable orientation, 2D X-ray diffraction patterns show spots that can be attributed to individual grains in the irradiated volume. Each reflection that the grains of a specific phase generate with a specific set of $\{hkl\}$ planes in that phase end up as a spot somewhere on a ring in the diffraction pattern. The radius of the ring characterizes the spacing d_{hkl} of the $\{hkl\}$ planes. Non-uniform distribution of spot intensities contains information about the grain size distribution, but convoluted with intensity variations owing to not perfectly fulfilling of the Bragg condition (1). Non-uniform distribution of spot locations over the circumference of the ring may

also arise from texture, i. e. non-uniform distribution of the grains' crystal lattice orientations. Elliptical deformation of rings indicates anisotropy of plane spacings, thus the presence of strain.

Several advantages can be combined when 2D X-ray diffraction patterns are recorded on a synchrotron beamline. As synchrotrons produce X-rays of high photon energy, patterns can be obtained *in transmission* even from macroscopic specimens, providing information on the *interior* of the material. Another advantage is the high degree of coherence (small photon energy spread, small beam convergence) of the synchrotron radiation. Accordingly, 2D diffraction patterns obtained by synchrotron beamline X-ray diffractometry (SXRD) are especially rich in information, enabling easier and more sensitive quantifcation of e. g. texture and strain.

While 2D diffraction patterns can contain information about individual grains, they are rarely analyzed spot by spot, owing to the large number and overlap of diffraction spots. Current XRD analysis techniques rely on laborious manual operations and can be time-consuming, especially when it comes to evaluating video sequences of patterns [1, 2]. For the conventional approach, therefore, more effectively analyzing 2D transmission diffraction patterns presents a significant challenge.

A solution to these challenges may be offered by *deep learning*, a state-of-the-art technique in artificial intelligence [3–6]. Deep learning focuses on building models that mimic the brain's ability to learn from data, which is a type of machine learning that uses algorithms to automatically analyze useful features from a data set. However, there is still a lack of exploration into the use of deep-learning techniques on 2D X-ray diffraction patterns [7, 8]. We hypothesize that by training deep-learning models on 2D X-ray diffraction patterns, it will be possible to automate and accelerate pattern analysis and produce highly accurate predictions about materials microstructures.

Our goal was to demonstrate this concept. In this study, we focus on using deep-learning techniques to predict the volume fraction of β-phase of titanium in the widely used Ti–6Al–4V alloy based on 2D SXRD diffraction patterns. In particular, we were interested in studying how heat treatment changes the volume fraction of β-phase.

As our deep-learning models, we employ convolutional neural networks (CNNs) [9, 10]. To enhance our models' performance while also fine-tuning them through the utilization of diverse model architectures, we employ image pre-processing techniques. [11–13]. Well-tuned models not only provide a higher level of accuracy compared to classical XRD analysis methods, but also exhibit greater robustness. They can effectively handle large volumes of XRD data sets, thereby reducing the reliance on manual operations.

1.1 Phase Identification in Ti–6Al–4V

Figure.1 shows a schematic of the SXRD experiment [14]. A monochromated incident X-ray beam is transmitted through a sample of Ti–6Al–4V, contained within an austenitic stainless steel (SS) container. The diffracted X-rays are captured by a two-dimensional digital detector, with 2048^2 pixels. According to Eq (1), the diffraction angle θ is the angle between the incident X-ray beam and the lattice plane. So the total angle by which the primary beam is diffracted is 2θ.

Predominantly, the Ti–6Al–4V sample consists of α-phase (hexagonal close-packed, HCP) and β-phase (body-centered cubic, BCC). The α-phase undergoes a transformation to the β-phase with increasing temperature. SXRD diffraction patterns consisting of these two phases (and the SS container) were collected at regular intervals (11 s apart) as the material was heated or cooled, allowing us to track changes in the crystal structure of the material as a function of time and temperature.

Each pixel in the SXRD pattern can be represented by the pixel coordinates (x, y) and the intensity i of the pixel, as well as the sequence order t of the pattern.

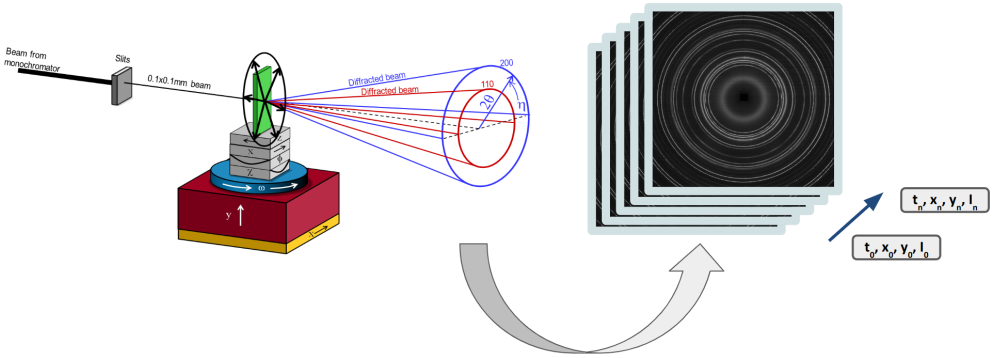


Fig. 1: Experimental procedure (figure modified from [14]). The experiment involved generating time-series SXRD diffraction patterns of the sample by directing a beam of X-rays onto it as it underwent heating and cooling.

Time-series diffraction patterns from the beginning to the end of the heat-treatment exhibit variation in the intensity of the sets of rings that represent α and β, owing to the change of α- and β-phase volume fractions with temperature (see Fig. 2).

The sample was held in a container of stainless steel, the walls of which were also penetrated by the primary and diffracted X-ray beams. Therefore, the SXRD diffraction patterns also exhibit spots from austenitic stainless steel. When calculating the phase fractions, the contributions from stainless steel are ignored.

1.2 Deep Learning and CNN Approach

Deep learning relies heavily on neural networks, which consist of multiple layers composed of individual neurons or perceptrons. Data is input into the neural network via an input layer. The learned results are obtained from an output layer. Any layers between the input and output layer are considered hidden layers, making the model a multi-layered perceptron (MLP) [15].

Furthermore, each neuron can be treated as a linear regression model where x is the input, w is the weight, and b is the bias:

$$y = wx + b \tag{2}$$

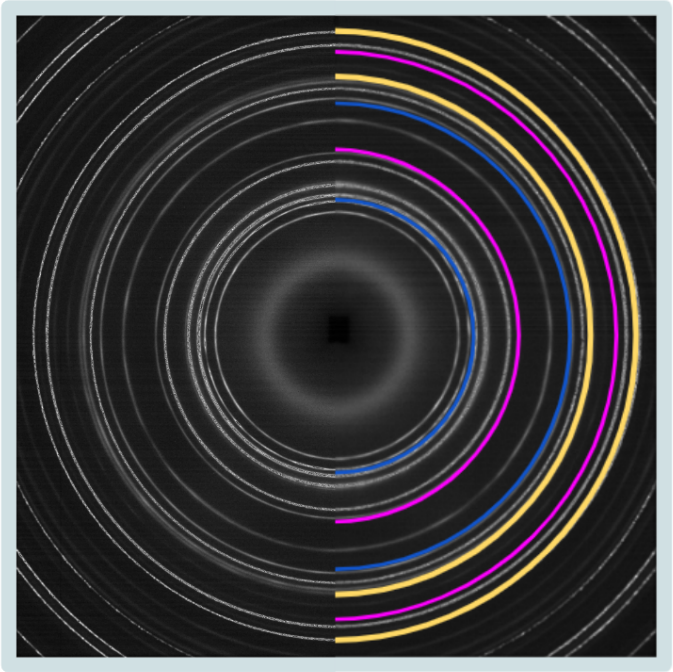


Fig. 2: Diffraction pattern from Ti–6Al–4V, exhibiting of different sets of rings. Each ring belongs to either the α-phase, β-phase, or austenitic stainless steel. The graphic shows examples of 2 rings for each category, with different colors: the pink rings correspond to the β-phase, the blue rings correspond to the α-phase, while the yellow rings represent austenitic stainless steel.

The output of a neuron y will be fed into neurons in the next layer until the model computes the final results in its output layer.

The quality of the response is measured via a loss function, which represents the discrepancy between final outputs of the model and actual values of the target feature for the object (i.e. the β-phase volume fraction for the input SXRD pattern). In order to minimize the discrepancy, i. e. find the minimum of the loss function, it is common to use gradient-descent techniques [16, 17] to tune parameters of the model during iterations.

In the past 10 years, CNNs and their varieties may have been the strongest deep-learning models in the computer-vision area. This is because the specific data structure of image data is very suitable for CNN analysis [18].

The convolutional layers stand out as one of the most distinctive characteristics of CNNs. These layers are designed to generate multiple small matrices (kernels), based on the dimensionality of the input data. Each kernel slides over the input data and performs a matrix product at each location. As the kernel completes a full pass over the input data, it generates a new pixel matrix, which is called a feature map. All feature maps are then passed on to the next layer. This process effectively extracts

Table 1: **Summary of SXRD data sets**

	Sample Names	# 2D Patterns	$T_{\max}$ (K)
Labeled	PB-HT1 (PB1)	1078	1043
	PB-HT2 (PB2)	1102	1113
	PB-HT3 (PB3)	863	1281
	WR-HT2 (WR2)	1103	1113
Unlabeled	WR-HT1 (WR1)	1079	1043
	WR-HT3 (WR3)	1057	1281
	LENS-HT1 (LENS1)	963	1043
	EBM-HT1 (EBM1)	1082	1043
	EBM-HT2 (EBM2)	1080	1113
	EBM-HT3 (EBM3)	880	1281

feature information from objects within the image and reduces the number of model parameters, which facilitates smoother model training.

After the convolutional layer, pooling layers are commonly employed to downsample the feature maps produced by the convolutional operation [19, 20]. Pooling layers aggregate nearby values in the feature maps by selecting specific representative values or computing statistics, such as the maximum or average value, from a local neighborhood. By downsampling the feature maps, pooling layers help to reduce the computational complexity of the network.

Finally, the output of these layers will be flattened into a 1D array and fed into the MLP, also called fully-connected (FC) or dense layers.

2 Experimental and Analytical Details

2.1 Data Set Collection

In order to observe how heat treatment induces microstructural changes in Ti–6Al–4V alloys with different microstructures, we studied four different kinds of alloy specimens:

- WR: Wrought alloy, obtained by cold working.
- PB: Alloy made by laser powder-bed fusion.
- LENS: Alloy made by laser engineered net shaping.
- EBM: Alloy made by electron-beam melting.

While wrought alloys are produced by conventional manufacturing, the latter three specimen types resulted from new techniques of *additive* manufacturing.

From each alloy, SXRD diffraction patterns were collected utilizing *in situ* high-energy XRD at the Cornell High-Energy Synchrotron Source (CHESS). Table 1 summarizes 10 experimental data sets collected during different thermal cycles in order to quantify the evolution of the β-phase volume fraction. Other experimental details, including diffraction conditions, are provided in a previous publication [14].

HT1, HT2, and HT3 are heat treatments conducted at 1043, 1113, and 1281 K, respectively. 2D diffraction patterns were collected with an integration time of 1 s every 10 s throughout the entire cycle of thermal treatment. Each specimen was heated from room temperature (300 K) to the maximum temperature at a rate of 100.0 K/min, held for 2.0 hours, and then cooled back down to room temperature at the same rate of 100.0 K/min. Four of the data sets (PB1, PB2, PB3, and WR2) have information about the β-phase volume fraction, which was determined by Rietveld refinement – implemented in GSAS-II [21] [14]. The β-phase volume fraction in the diffraction patterns of the labeled dataset is known, but we have no corresponding information about unlabeled data sets. Therefore, we utilize these labeled data sets to train CNN models that can automatically predict the β-phase volume fraction with high accuracy for any Ti–6Al–4V 2D XRD diffraction patterns.

2.2 Image Pre-processing

Before feeding the 2D SXRD data into CNN models, we pre-processed diffraction patterns by applying dark-current correction, pattern centering, and pattern re-scaling (see Fig 3).

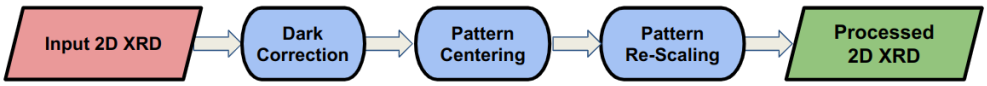


Fig. 3: The pipeline that shows image pre-processing steps, include dark-current correction, pattern centering and pattern re-scaling

Dark-Current Correction

"Dark current" refers to the apparent signal produced by the detector even when no X-rays impinge on it. The corresponding background needs to be subtracted, as it can interfere with the proper evaluation of the true X-ray signal [22]. To remove the dark-current background from each SXRD pattern, we subtract the intensities of a reference "dark" pattern, provided by our collaborator, from the input patterns. Each experimental data set utilizes a single unique reference "dark" pattern, and all input patterns within each data set are processed using the same reference dark pattern.

Pattern Centering

Before CNN analysis, we ensure that the SXRD patterns are perfectly centered. To achieve this, we construct a mask consisting of 50 concentric rings as shown in Fig 4 with varying radii at the center of the pattern. Next, we slightly move the center of these concentric rings within a small range around the pattern center, confined within a bounding box with dimensions of 40 by 40 pixels. For each configuration, we multiply the mask with the input pattern and record the sum of pixel values for output pattern. When the center of the concentric rings aligns with the center of the rings in the input pattern, the output pattern will exhibit the highest sum of pixel intensities.

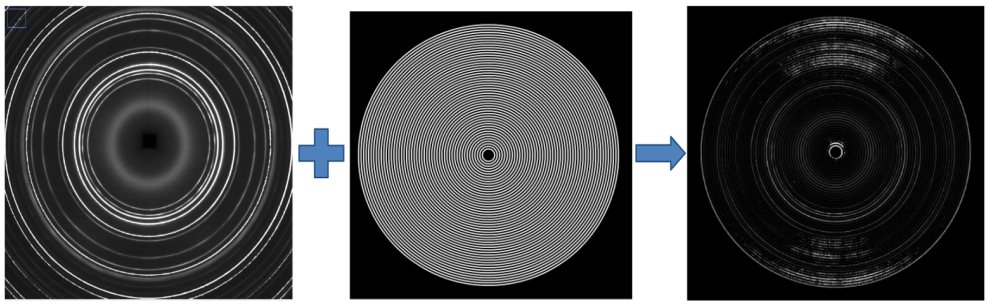


Fig. 4: Image pre-processing. Left: Original diffraction pattern. Middle: Mask with 50 centered concentric sampling rings, 8 pixels wide radii spaced 10 pixels apart. Radius of the innermost ring: 40 pixels. Right: Result of multiplying the mask with the original pattern.

Once we identified the coordinates of the center of the rings in a SXRD pattern, we employed a shifting procedure to align this center with the pattern center. To test whether a shift of two pixels to the left was required, we remove two columns from the left side of the pixel matrix representing the pattern. Similarly, we crop the top or bottom rows to shift the center of the rings upwards or downwards, respectively. Once the rings' center is successfully shifted to the pattern center, we zoom the cropped pattern to its original dimensions.

Pattern Re-Scaling

During SXRD experiments, all samples are mounted in a austenitic stainless steel container, so their SXRD patterns also exhibit diffraction rings from austenitic stainless steel.

For reasons explained in Section 4.1, we re-scaled each pattern such that the diameters of its rings from stainless steel match those observed in a reference ring pattern from stainless steel at room temperature (300 K). To achieve this, we iteratively scale the pattern at hand with different ratios and compute the pixel-wise correlation [23] between each resized pattern and the reference pattern. As higher correlation indicates a greater pattern similarity, the scaled pattern that produces the highest pixel-wise correlation value is considered to be correctly scaled. In the following, we call this operation "pattern re-scaling."

Quality Analysis

In order to assess the quality of our image pre-processing, we computed pixel-wise correlations between a reference pattern, consisting of centered stainless steel rings at room temperature only, and both the raw and the processed patterns obtained from data sets PB1, PB2, and PB3. (Higher correlation indicates a greater pattern similarity.)

2.3 CNN Implementation

Our CNN models were trained on a total of 3008 SXRD patterns, which consisted of PB1, PB2 and PB3 labeled data sets. These 3008 diffraction patterns are split into a training set comprising 65 % of the data, a validation set comprising 15 % of the data, and a test set comprising 20 % of the data.

CNN models were implemented using the Tensorflow framework. To limit pixel intensities to values between 0 and 1, the pixel intensities of each SXRD patterns were divided by a scaling factor. To assess the accuracy of the model, we computed the mean squared error (MSE) by calculating the average of the squared differences between the predicted values and the actual values. The training was terminated when there was no improvement in the MSE on the validation set for 7 consecutive epochs. After training these models, we used them to predict the volume fraction of the β-phase for WR2 data sets.

Architecture of CNN model is decided by multiple hyper-parameters. We also trained various CNN models with different combinations of these hyper-parameters [13, 24–26]. They include number of convolutional layers, the number of fully connected layers, whether or not implement batch normalization and dropout techniques, as well as type of regularization employed. Each of these in our model plays a unique role in both training and performance outcome. The number of layers and the number of nodes within each layer are essential factors that influence the model's capability to learn complex patterns. Generally, a model with more layers can effectively handle larger amounts of data and capture more intricate features within the data. However, increasing the number of layers also leads to higher training costs. A model that is too complex can lead to overfitting, where the model performs well on the training set but fails to generalize to new, unseen data. Dropout layers are effective in reducing overfitting by randomly deactivating a portion of the neurons within the hidden layers during training [33, 34]. Batch normalization layers normalize the outputs of intermediate layers during training, ensuring that they have a consistent and limited range. This normalization aids in stabilizing the learning process and promoting faster convergence [12, 35]. L1-L2 regularization will add an additional term to the loss function to impose constraints on the weights of the model. By promoting sparse weight distributions, it enhances the model's generalization ability, resulting in a more robust and capable model [11, 36].

In order to optimize these hyper-parameters for the CNN model, we created a pipeline (shown in Fig 5) that automates the tuning process. Users can input a range of values for each hyper-parameter, and our package will generate a CSV file containing all possible combinations within those ranges. For each combination, the package will generate a model and initiate a training job on our high-performance computing (HPC) system, which is equipped with the necessary GPU, CPU, and memory.

This is achieved by submitting the job through a tool known as Linux Utility for Resource Management (SLURM) [27, 28]. We also embedded SLURM scripts into our Python script, which we named **sdleSlurm**. This enabled us to easily perform model creation, hyper-parameter tuning, and parallel training within the Python environment, ensuring effective code maintainability.

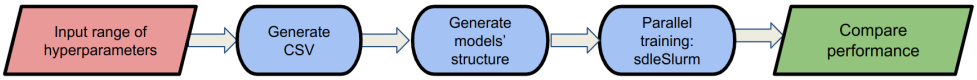


Fig. 5: A pipeline that shows automatically adjusts the settings of CNN based on the user's preferred range of hyper-parameters.

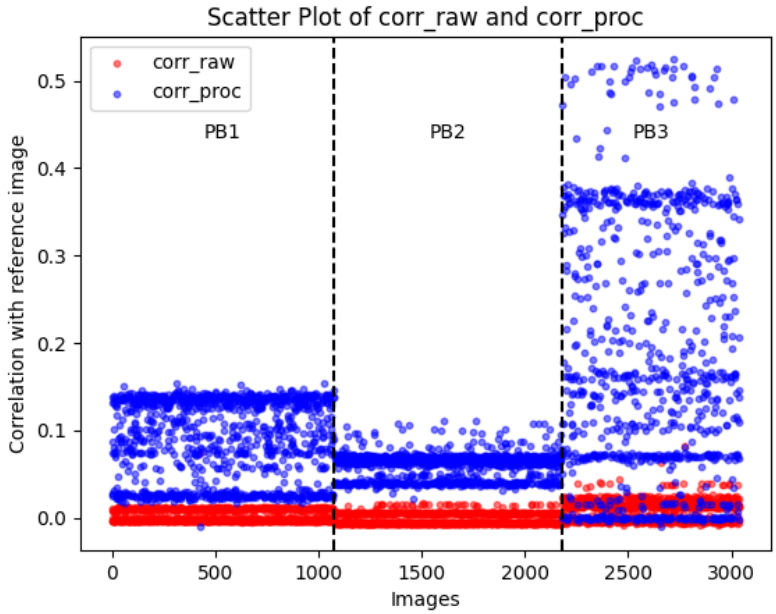


Fig. 6: Scatter plots for samples PB1, PB2, and PB3 indicating pixelwise correlation between (i) raw pattens and reference pattern (red) and (ii) processed patterns and reference pattern (blue).

3 Results

Regarding image pre-processing, Fig. 6 consistently shows – for PB1, PB2 and PB3 – that pre-processing increases the correlation (similarity) with the reference pattern.

In total, 168 CNN models were trained, using various hyper-parameters. The results for all models are compiled in Table A1, A2, A3, A4 and A5 in Appendix A. The tables present the following information:

1. Architecture of each model. Specifically, the tables include details such as the number of convolutional layers (#CL), which ranges from 4 to 12, the number of fully connected layers (#Dense), which ranges from 2 to 3, and the count of nodes in each fully connected layer.

2. Model performance, via the MSE observed for training, validation, testing sets, and WR2 data sets.
3. Whether or not batch normalization (BN) and dropout techniques were implemented, denoted by "True" and "False," respectively.
4. Type of regularization employed, indicated by one with four possible values: "1" for L1-regularization, "2" for L2-regularization, "3" for L1-L2 regularization, and "0" indicating the absence of regularization. The rationale for using these hyperparameters is described in section 4.2.

Model #80, which includes 9 convolution layers and 2 fully connected layers, achieved the lowest MSE among all 168 models on the WR2 data set: 2.6×10^{-4}. Only for the SXRD patterns recorded during the initial part of the heat-treatment, Model #80 does not generate accurate predictions. Figure 7 visualizes the architecture of Model #80. Dropout layers with a rate of 0.50 were applied to the model. In total, 124,957,233 parameters were used. Figure 8 illustrates the training process and displays the predicted volume fraction of β-phase for each SXRD pattern in data sets PB1, PB2, PB3, and WR2.

During the training process of Model #80, we noticed that this model quickly reaches stability and converges within just 2 epochs. However, after 12 epochs, we observed an increase in the loss for the validation sets. To prevent overfitting and ensure optimal generalization, an early stop mechanism was implemented. It halted the training after 15 epochs.

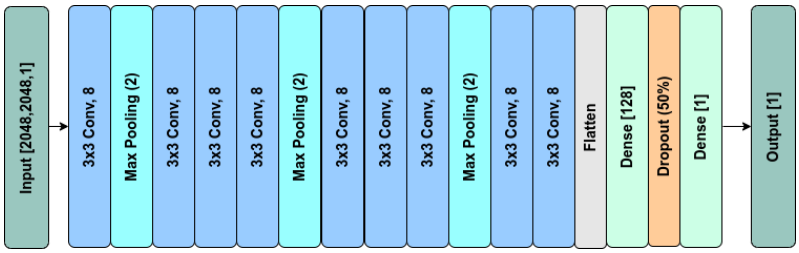
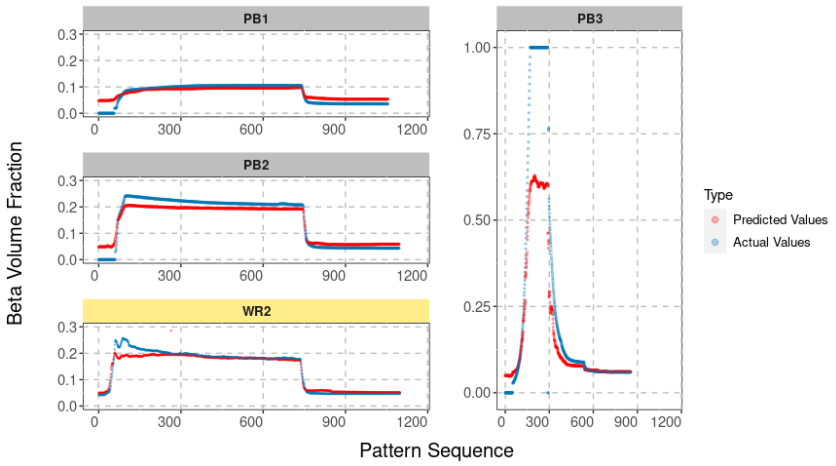


Fig. 7: Architecture of CNN Model #80, comprised of 9 convolutional layers and 2 dense layers.

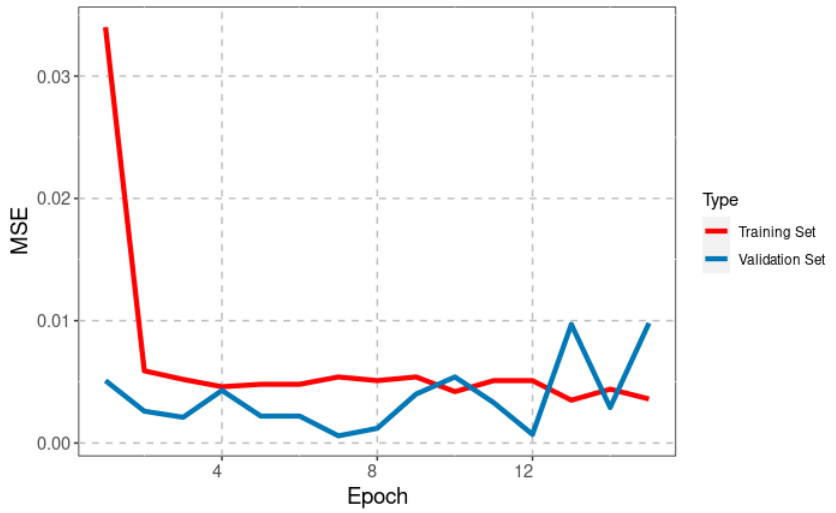
4 Discussion

4.1 Analysis of Image Pre-processing

One particular step of our pre-processing procedure turned out to be crucial: *pattern centering*. In the past, it was believed that CNNs could detect objects in different positions within an image because the convolutional filters applied to the input image would shift the whole image, allowing the features of the objects to be detected regardless of their position. This property of CNNs is known as "translation invariance."



(a) Predicted results of Model #80 on PB1, PB2, PB3 and WR2 data sets



(b) Training History of Model #80

Fig. 8: Model #80. (a) Actual (blue) and predicted (red) values on all labeled data sets, including PB1, PB2, PB3, and WR2. (b) Training history, terminated after 15 epochs by early stopping.

However, consistent with our present observations, recent research challenged this assumption [29–32].

Further, we found that not all SXRD patterns actually contain intense rings from stainless steels. Therefore, if our models are influenced by stainless steel rings, its capability may be compromised in scenarios where these rings are not present with sufficient intensity. Instead, our pattern re-scaling, serves to ensure that the rings have

a consistent size. In fact, the expansion of the stainless steel rings, α-phase rings and of the β-phase rings are different during the heat-treatment. So, re-scaling our pattern with the same ratio would inevitably reduce data quality. Nevertheless, the rings of stainless steel exhibit significantly higher intensity compared to others. This implies that the model will allocate greater attention to these features. So, re-scaling them can be considered a fair trade-off to mitigate their influence.

After pattern pre-processing, it is expected that the stainless steel rings within each pattern will have the same diameter and the same center as in the reference pattern. Accordingly, their pixel-wise correlation with the reference should be higher than before pre-processing. The fact that Fig. 6 confirms this expectation indicates that our image pre-processing techniques are highly effective in transforming the patterns such that their stainless steel rings closely resemble those observed at room-temperature.

4.2 Performance of CNN Models in SXRD Analysis

We strived to find a model that minimizes the MSE, corresponding to minimal discrepancy between the predicted results and the actual values. Our primary focus was on evaluating the performance of the models on the WR2 data set, as this data set was not included during the training phase. The MSE on the WR2 data set served as a reliable metric to assess the capabilities of the models. By analyzing it, we can gain insights into how well the models generalize to unseen data and effectively gauge their overall performance. Model #80 outperformed others on WR2 data sets with an MSE of 2.6×10^{-4}. So, the average prediction error for the β-phase fraction – in each diffraction pattern of WR2 – was within 1.6×10^{-2} of the values obtained by conventional methods. However, Model #80 displayed relatively poor performance on the testing data sets. It recorded a MSE of 6.4×10^{-3} (while achieving a MSE of 3.6×10^{-3} on the training set).

Further examination of the performance characteristics of Model #80 are shown in Fig. 7, which depicts the model's predicted β-phase fraction juxtaposed with the experimental values, along with the training history of the model. As observed from in Fig. 8a, a significant disparity between the predicted and actual values can be observed in the PB3 data sets, even though PB3 is a subset of the training set. Consequently, the bad MSE for the testing set (6.4×10^{-3}) can be attributed to this disparity. To be more specific, the PB3 data set contains some SXRD patterns characterized by a high volume fraction of β-phase, approaching 1.00. However, the model struggled to produce accurate predictions for these particular SXRD patterns.

Since our goal is to identify models that exhibit robustness, meaning that they can perform consistently well across all data sets, we present the MSE results in Table 2, including the top 15 models for the WR2 and the testing data set. Models that perform well on the testing data set indicate that they are well-trained. Nevertheless, not all well-trained models will also perform well on external data sets, such as WR2. In fact, out of the top 15 models, only 9 have demonstrated strong performance in both cases (Model #12, 16, 28, 32, 35, 36, 40, 44, 8). The architecture of these models is summarized in Table 3.

Figure 9 shows the performance of the models. All four data sets are combined in the order of PB1, PB2, PB3, and WR2. The 9 models consistently perform well,

Table 2: Top 15 CNN models ranked by their MSE on both the test and WR2 datasets

Model#	MSE-test	Model#	MSE-WR2
16	3.9×10^{-4}	80	2.5×10^{-4}
32	4.3×10^{-4}	38	4.4×10^{-4}
8	4.9×10^{-4}	44	4.7×10^{-4}
36	5.0×10^{-4}	35	4.9×10^{-4}
40	5.1×10^{-4}	12	5.6×10^{-4}
28	5.8×10^{-4}	16	6.6×10^{-4}
48	6.2×10^{-4}	36	7.2×10^{-4}
12	6.3×10^{-4}	2	7.8×10^{-4}
44	6.6×10^{-4}	46	7.9×10^{-4}
4	6.8×10^{-4}	28	8.4×10^{-4}
14	8.8×10^{-4}	40	10×10^{-4}
30	9.6×10^{-4}	32	12×10^{-4}
148	9.8×10^{-4}	8	13×10^{-4}
35	12×10^{-4}	4	14×10^{-4}
24	13×10^{-4}	148	15×10^{-4}

without showing poor predictions for any data set (unlike Model #80). Therefore, we consider these nine models to be more robust options for the phase identification task at hand. We also noticed that some predicted values exceed 1, approaching 1.1. Indeed, we chose not to enforce constraints that would limit all predicted values to be between 0 and 1, as adding constraints within the loss function will increase the complexity during training, affecting model's convergence and flexibility. Moreover, without constraints, the model can detect outliers, preventing them from being automatically mapped to the range between 0 and 1.

Next, after analyzing the hyper-parameters of the 9 models presented in Table 9, we observe that none of them utilized L1-L2 regularization or dropout layers and only Model #35 utilized batch normalization. While these techniques are frequently employed to mitigate complexity and prevent overfitting, it seems that their incorporation did not yield notable performance advantages for our models. Additionally, we discovered that some of our models generated identical predictions for all SXRD patterns after training, particularly those with more than 8 convolutional layers. Based on our findings, it seems that more complex models do not necessarily result in higher accuracy and may even fail to learn from the data. It also proves that the architecture of model plays a critical role in our XRD analysis tasks, highlighting the importance of fine-tuning hyper-parameters [13].

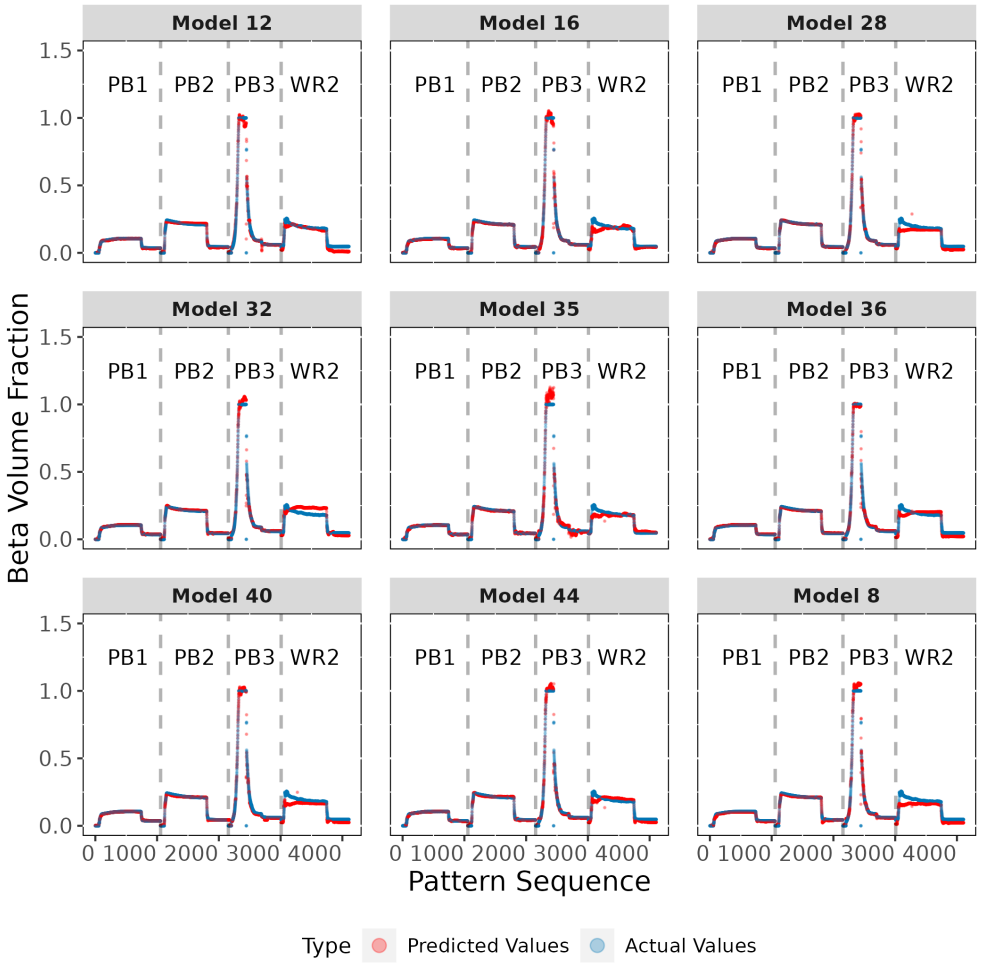


Fig. 9: Actual versus predicted values for 9 models. Every sub-image represents one model. Each block divided by a dashed line within the image represents just one data set.

4.3 Deep Learning Versus Conventional Methods in XRD Analysis

There are several conventional methods to identify phases and quantify their volume fractions. For 2D XRD data, one way is to apply rotation averaging to reduce them to 1D diffraction patterns, where the y-axis represents diffracted intensity and the x-axis represents the spatial frequency q, i. e. the inverse spacing of the diffracting planes. Each peak in the 1D pattern corresponds to specific $\{hkl\}$ crystal planes in the sample. A typical conventional evaluation would first compute the positions and

Table 3: 9 robustness models along with their hyper-parameters and performance metrics

Index	MSE-test	MSE-WR2	#CL	#Dense	Dropout	BN	L1-L2-Reg
8	4.9×10^{-4}	12×10^{-4}	4	128-64	F	F	0
12	6.3×10^{-4}	5.6×10^{-4}	4	128-64-32	F	F	0
16	3.9×10^{-4}	6.6×10^{-4}	5	128	F	F	0
28	5.9×10^{-4}	8.4×10^{-4}	6	128	F	F	0
32	4.3×10^{-4}	12×10^{-4}	6	128-64	F	F	0
35	13×10^{-4}	4.9×10^{-4}	6	128-64-32	F	T	0
36	5.0×10^{-4}	7.2×10^{-4}	6	128-64-32	F	F	0
40	5.1×10^{-4}	10×10^{-4}	7	128	F	F	0
44	6.6×10^{-4}	4.7×10^{-4}	7	128-64	F	F	0

relative intensities of the peaks, then comparing them with those of experimental or simulated reference patterns from known crystal structures of various phases [1, 2].

An alternative method involves utilizing software like GSAS-II to generate simulated SXRD patterns by inputting material parameters, such as grain size, texture, etc [14]. The software can then compare the simulated patterns to experimental SXRD patterns and "refine" e. g. the phase fractions by optimizing the match between the theoretical and experimental pattern.

In contrast, our trained deep-learning model can provide volume fraction of phases directly. Compared to 1D patterns, individual spots that appear in 2D patterns contain additional information, encoded in their number, intensity, and position. Given that our model utilizes 2D patterns as input, we conclude that it effectively learns and captures these important details [22].

During the training process of the model, the label information generated by conventional methods is essential. However, once the model is sufficiently trained, it can be applied to analyze millions of XRD patterns, e. g. provided as video sequences leading to *in-situ* analysis at beamlines. Moreover, as the amount of labeled data increases, the accuracy of the model continuously improves.

5 Conclusions

While 2D XRD diffraction patterns contain more detailed information about materials than 1D XRD patterns, effectively tracking and analyzing this information poses more complex challenges. For the example of determining the volume fraction of β-phase in Ti–6Al–4V, our research demonstrates the effectiveness of CNN models. We observed significant performance variation of these models on different data sets, allowing us to identify the top-performing models consistently across both data sets. These models achieved a high level of accuracy when compared to conventional analysis methods. However, the highest accuracy is not necessarily achieved by the most complex models. Those may even fail to learn from the data. This emphasizes the importance of tuning

CNN models to ensure robustness and excellent performance across all Ti–6Al–4V SXRD data sets.

While our work demonstrates the potential of utilizing deep-learning models to assist performing phase-fraction analysis on SXRD patterns, its applicability is broader. In general, employing a well-trained CNN model provides efficient and automated analysis of 2D XRD patterns, delivering accurate results within a shorter timeframe. This approach offers several advantages, including the ability to handle large volumes of data, extract meaningful features, and make precise predictions. Leveraging the power of deep-learning techniques enables valuable insights into materials microstructures that were previously challenging to obtain.

Acknowledgments

This material is based upon research in the Materials Data Science for Stockpile Stewardship Center of Excellence (MDS3-COE), and supported by the Department of Energy's National Nuclear Security Administration under Award Number(s) DE-NA0004104. Abridged Legal Disclaimer: "The views expressed herein do not necessarily represent the views of the U.S. Department of Energy or the United States Government." This work made use of the High Performance Computing Resource in the Core Facility for Advanced Research Computing at Case Western Reserve University. This article was made possible, in part, through financial support from the School of Graduate Studies at Case Western Reserve University. This work is based on research conducted at the Center for High-Energy X-ray Sciences (CHEXS), which is supported by the National Science Foundation (BIO, ENG and MPS Directorates) under award DMR-1829070. This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

References

[1] Chauhan, A.: Powder XRD Technique and its Applications in Science and Technology. Journal of Analytical & Bioanalytical Techniques **5**(6) (2014). https://doi.org/10.4172/2155-9872.1000212. Accessed 2023-04-27

[2] Rodriguez-Navarro, C., Kudlacz, K., Ruiz-Agudo, E.: The mechanism of thermal decomposition of dolomite: New insights from 2D-XRD and TEM analyses. American Mineralogist **97**(1), 38–51 (2012). https://doi.org/10.2138/am.2011.3813

[3] Mahony, N.O., Campbell, S., Carvalho, A., Harapanahalli, S., Velasco-Hernandez, G., Krpalkova, L., Riordan, D., Walsh, J.: Deep Learning Vs. Traditional Computer Vision vol. 943, (2020). https://doi.org/10.1007/978-3-030-17795-9. arXiv:1910.13796 [cs]. http://arxiv.org/abs/1910.13796 Accessed 2023-04-27

[4] Deng, L., Yu, D.: Deep Learning: Methods and Applications. Foundations and Trends® in Signal Processing **7**(3–4), 197–387 (2014). https://doi.org/10.1561/

2000000039

[5] Choudhary, K., DeCost, B., Chen, C., Jain, A., Tavazza, F., Cohn, R., Park, C.W., Choudhary, A., Agrawal, A., Billinge, S.J.L., Holm, E., Ong, S.P., Wolverton, C.: Recent advances and applications of deep learning methods in materials science. npj Computational Materials **8**(1), 1–26 (2022). https://doi.org/10.1038/s41524-022-00734-6

[6] Oviedo, F., Ren, Z., Sun, S., Settens, C., Liu, Z., Hartono, N.T.P., Ramasamy, S., DeCost, B.L., Tian, S.I.P., Romano, G., Gilad Kusne, A., Buonassisi, T.: Fast and interpretable classification of small X-ray diffraction datasets using data augmentation and deep neural networks. npj Computational Materials **5**(1), 1–9 (2019). https://doi.org/10.1038/s41524-019-0196-x

[7] Ross, K.C., Petrus, J.A., McDonald, A.M.: An empirical assessment of two-dimensional (2D) Debye–Scherrer-type image-plate X-ray diffraction data collapsed into a 1D diffractogram. Powder Diffraction **29**(4), 337–345 (2014). https://doi.org/10.1017/S0885715614000487. Publisher: Cambridge University Press. Accessed 2023-04-27

[8] Xiong, Y., Xie, Y., Du, G., Su, H.: From 2D Framework to Quasi-1D Nanomaterial: Preparation, Characterization, and Formation Mechanism of Cu3SnS4 Nanorods. Inorganic Chemistry **41**(11), 2953–2959 (2002). https://doi.org/10.1021/ic0200242

[9] Potluri, S., Fasih, A., Vutukuru, L.K., Machot, F.A., Kyamakya, K.: CNN based high performance computing for real time image processing on GPU. In: Proceedings of the Joint INDS'11 & ISTET'11, pp. 1–7 (2011). https://doi.org/10.1109/INDS.2011.6024781

[10] Review of deep learning: concepts, CNN architectures, challenges, applications, future directions | SpringerLink. https://link.springer.com/article/10.1186/s40537-021-00444-8 Accessed 2023-04-27

[11] Demir-Kavuk, O., Kamada, M., Akutsu, T., Knapp, E.-W.: Prediction using stepwise L1, L2 regularization and feature selection for small data sets with large number of features. BMC Bioinformatics **12**, 412 (2011). https://doi.org/10.1186/1471-2105-12-412. Accessed 2023-04-27

[12] Ismail, A., Ahmad, S.A., Soh, A.C., Hassan, K., Harith, H.H.: Improving Convolutional Neural Network (CNN) architecture (miniVGGNet) with Batch Normalization and Learning Rate Decay Factor for Image Classification. International Journal of Integrated Engineering **11**(4) (2019)

[13] Yu, T., Zhu, H.: Hyper-Parameter Optimization: A Review of Algorithms and Applications. arXiv. arXiv:2003.05689 [cs, stat] (2020). https://doi.org/10.48550/arXiv.2003.05689. http://arxiv.org/abs/2003.05689 Accessed 2023-04-27

[14] Brown, D.W., Anghel, V., Balogh, L., Clausen, B., Johnson, N.S., Martinez, R.M., Pagan, D.C., Rafailov, G., Ravkov, L., Strantza, M., Zepeda-Alarcon, E.: Evolution of the Microstructure of Laser Powder Bed Fusion Ti-6Al-4V During Post-Build Heat Treatment. Metallurgical and Materials Transactions A: Physical Metallurgy and Materials Science **52**(12), 5165–5181 (2021). https://doi.org/10.1007/s11661-021-06455-7. Accessed 2023-04-27

[15] Taud, H., Mas, J.F.: Multilayer Perceptron (MLP). In: Camacho Olmedo, M.T., Paegelow, M., Mas, J.-F., Escobar, F. (eds.) Geomatic Approaches for Modeling Land Change Scenarios. Lecture Notes in Geoinformation and Cartography, pp. 451–455. Springer International Publishing, Cham (2018). https://doi.org/10.1007/978-3-319-60801-3_27

[16] Zhang, C., Yao, M., Chen, W., Zhang, S., Chen, D., Wu, Y.: Gradient Descent Optimization in Deep Learning Model Training Based on Multistage and Method Combination Strategy. Security and Communication Networks **2021**, 9956773 (2021). https://doi.org/10.1155/2021/9956773. Publisher: Hindawi. Accessed 2023-04-27

[17] Haji, S.H., Abdulazeez, A.M.: COMPARISON OF OPTIMIZATION TECHNIQUES BASED ON GRADIENT DESCENT ALGORITHM: A REVIEW. PalArch's Journal of Archaeology of Egypt / Egyptology **18**(4), 2715–2743 (2021)

[18] Chauhan, R., Ghanshala, K.K., Joshi, R.C.: Convolutional Neural Network (CNN) for Image Detection and Recognition. In: 2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC), pp. 278–282 (2018). https://doi.org/10.1109/ICSCCC.2018.8703316

[19] Christlein, V., Spranger, L., Seuret, M., Nicolaou, A., Král, P., Maier, A.: Deep Generalized Max Pooling. In: 2019 International Conference on Document Analysis and Recognition (ICDAR), pp. 1090–1096 (2019). https://doi.org/10.1109/ICDAR.2019.00177

[20] Nagi, J., Ducatelle, F., Di Caro, G.A., Cireşan, D., Meier, U., Giusti, A., Nagi, F., Schmidhuber, J., Gambardella, L.M.: Max-pooling convolutional neural networks for vision-based hand gesture recognition. In: 2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA), pp. 342–347 (2011). https://doi.org/10.1109/ICSIPA.2011.6144164

[21] Toby, B.H., Von Dreele, R.B.: GSAS-II: The genesis of a modern open-source all purpose crystallography software package. Journal of Applied Crystallography **46**(2), 544–549 (2013). https://doi.org/10.1107/S0021889813003531

[22] Vallcorba, O., Rius, J.: d2Dplot: 2D X-ray diffraction data processing and analysis for through-the-substrate microdiffraction. Journal of Applied Crystallography **52**(2), 478–484 (2019). https://doi.org/10.1107/S160057671900219X

[23] Bosse, S., Maniry, D., Wiegand, T., Samek, W.: A deep neural network for image quality assessment. In: 2016 IEEE International Conference on Image Processing (ICIP), pp. 3773–3777 (2016). https://doi.org/10.1109/ICIP.2016.7533065

[24] A Tutorial on Automatic Hyperparameter Tuning of Deep Spectral Modelling for Regression and Classification Tasks - ScienceDirect. https://www.sciencedirect.com/science/article/pii/S0169743922000314

[25] Kimura, G.Y., Lucio, D.R., Britto Jr., A.S., Menotti, D.: CNN Hyperparameter Tuning Applied to Iris Liveness Detection. arXiv (2020). https://doi.org/10.48550/arXiv.2003.00833

[26] Prechelt, L.: Early Stopping - But When? In: Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. Lecture Notes in Computer Science, pp. 55–69. Springer, Berlin, Heidelberg (1998). https://doi.org/10.1007/3-540-49430-8_3. https://doi.org/10.1007/3-540-49430-8_3 Accessed 2023-04-27

[27] Yoo, A.B., Jette, M.A., Grondona, M.: SLURM: Simple Linux Utility for Resource Management. In: Feitelson, D., Rudolph, L., Schwiegelshohn, U. (eds.) Job Scheduling Strategies for Parallel Processing. Lecture Notes in Computer Science, pp. 44–60. Springer, Berlin, Heidelberg (2003). https://doi.org/10.1007/10968987_3

[28] Campos, V., Sastre, F., Yagües, M., Bellver, M., Giró-i-Nieto, X., Torres, J.: Distributed training strategies for a computer vision deep learning algorithm on a distributed GPU cluster. Procedia Computer Science **108**, 315–324 (2017). https://doi.org/10.1016/j.procs.2017.05.074

[29] Azulay, A., Weiss, Y.: Why do deep convolutional networks generalize so poorly to small image transformations? arXiv. arXiv:1805.12177 [cs] (2019). https://doi.org/10.48550/arXiv.1805.12177. http://arxiv.org/abs/1805.12177 Accessed 2023-04-27

[30] Zhang, R.: Making Convolutional Networks Shift-Invariant Again. In: Proceedings of the 36th International Conference on Machine Learning, pp. 7324–7334. PMLR, (2019)

[31] Kayhan, O.S., van Gemert, J.C.: On Translation Invariance in CNNs: Convolutional Layers Can Exploit Absolute Spatial Location. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 14274–14285 (2020)

[32] Furat, O., Wang, M., Neumann, M., Petrich, L., Weber, M., Krill, C.E., Schmidt, V.: Machine Learning Techniques for the Segmentation of Tomographic Image Data of Functional Materials. Frontiers in Materials **6** (2019)

[33] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.:

Dropout: A Simple Way to Prevent Neural Networks from Overfitting

[34] Wu, H., Gu, X.: Towards dropout training for convolutional neural networks. Neural Networks **71**, 1–10 (2015). https://doi.org/10.1016/j.neunet.2015.07.007

[35] An Overview of Overfitting and its Solutions - IOPscience. https://iopscience.iop.org/article/10.1088/1742-6596/1168/2/022022/meta Accessed 2023-04-27

[36] Farhadi, Z., Bevrani, H., Feizi-Derakhshi, M.-R.: Combining Regularization and Dropout Techniques for Deep Convolutional Neural Network. In: 2022 Global Energy Conference (GEC), pp. 335–339 (2022). https://doi.org/10.1109/GEC55014.2022.9986657

Appendix A

Table A1: Model 1–40

Index	MSE-train	MSE-val	MSE-test	MSE-WR2	#CL	#Dense	Dropout	BN	L1-L2-Reg
1	0.04098	0.05118	0.03625	0.00533	4	128	T	T	0
2	0.00400	0.00177	0.00175	0.00078	4	128	T	F	0
3	0.04270	0.05232	0.03067	0.00563	4	128	F	T	0
4	0.00092	0.00046	0.00068	0.00143	4	128	F	F	0
5	0.04121	0.05108	0.03135	0.00542	4	128-64	T	T	0
6	0.00486	0.00107	0.00159	0.00311	4	128-64	T	F	0
7	0.09729	0.03627	0.03160	0.04671	4	128-64	F	T	0
8	0.00022	0.00019	0.00049	0.00125	4	128-64	F	F	0
9	0.04268	0.11871	0.11549	0.00536	4	128-64-32	T	T	0
10	0.00745	0.00364	0.00321	0.00039	4	128-64-32	T	F	0
11	100.23955	35.24343	46.44252	6.04375	4	128-64-32	F	T	0
12	0.00038	0.00016	0.00063	0.00056	4	128-64-32	F	F	0
13	0.04107	0.05116	0.03036	0.00535	5	128	T	T	0
14	0.00418	0.00063	0.00088	0.00244	5	128	T	F	0
15	0.00342	0.00259	0.00231	0.00164	5	128	F	T	0
16	0.00026	0.00012	0.00039	0.00066	5	128	F	F	0
17	91.17519	8.07776	8.13475	0.06583	5	128-64	T	T	0
18	0.04098	0.05113	0.03044	0.00541	5	128-64	T	F	0
19	13.77550	3.52833	4.63186	20.37558	5	128-64	F	T	0
20	0.04111	0.05157	0.03033	0.00530	5	128-64	F	F	0
21	0.04115	0.05114	0.28407	0.00538	5	128-64-32	T	T	0
22	0.04096	0.05114	0.03043	0.00540	5	128-64-32	T	F	0
23	0.00272	0.00575	0.00587	0.09908	5	128-64-32	F	T	0
24	0.00188	0.00081	0.00132	0.00627	5	128-64-32	F	F	0
25	0.01595	0.00680	0.00414	0.02338	6	128	T	T	0
26	0.00329	0.00245	0.00286	0.04248	6	128	T	F	0
27	0.00779	0.00335	0.00371	0.14714	6	128	F	T	0
28	0.00019	0.00003	0.00059	0.00084	6	128	F	F	0
29	0.04101	0.05113	0.03043	0.00652	6	128-64	T	T	0
30	0.00475	0.00079	0.00096	0.00470	6	128-64	T	F	0
31	0.02199	0.01761	0.02222	0.01489	6	128-64	F	T	0
32	0.00029	0.00031	0.00043	0.00125	6	128-64	F	F	0
33	0.04104	0.05114	0.03041	0.00539	6	128-64-32	T	T	0
34	0.00940	0.00810	0.00570	0.01219	6	128-64-32	T	F	0
35	0.00117	0.00062	0.00125	0.00049	6	128-64-32	F	T	0
36	0.00023	0.00012	0.00051	0.00072	6	128-64-32	F	F	0
37	0.01816	0.00486	0.00456	0.00867	7	128	T	T	0
38	0.00301	0.00049	0.00158	0.00044	7	128	T	F	0
39	0.47298	0.51769	0.49785	0.74166	7	128	F	T	0
40	0.00027	0.00022	0.00051	0.00100	7	128	F	F	0

Table A2: **Model 41–80**

Index	MSE-train	MSE-val	MSE-test	MSE-WR2	#CL	#Dense	Dropout	BN	L1-L2-Reg
41	0.04382	0.05197	0.03197	0.00544	7	128-64	T	T	0
42	0.04098	0.05113	0.03044	0.00541	7	128-64	T	F	0
43	0.98697	0.04093	0.03905	0.00307	7	128-64	F	T	0
44	0.00029	0.00024	0.00066	0.00047	7	128-64	F	F	0
45	0.04095	0.05112	0.03048	0.00545	7	128-64-32	T	T	0
46	0.00585	0.00102	0.00217	0.00079	7	128-64-32	T	F	0
47	0.00162	0.00109	0.00162	0.00657	7	128-64-32	F	T	0
48	0.00024	0.00019	0.00062	0.00202	7	128-64-32	F	F	0
49	0.04097	0.05113	0.03120	0.00635	8	128	T	T	3
50	0.04097	0.05113	0.03127	0.00624	8	128	T	T	1
51	0.01617	0.01903	0.01880	0.00916	8	128	T	T	2
52	0.04096	0.05113	0.03047	0.00544	8	128	T	T	0
53	0.04098	0.05113	0.03044	0.00541	8	128	T	F	3
54	0.04098	0.05113	0.03045	0.00541	8	128	T	F	1
55	0.04098	0.05113	0.03044	0.00541	8	128	T	F	2
56	0.04100	0.05113	0.03043	0.00540	8	128	T	F	0
57	0.04098	0.05113	0.03077	0.00592	8	128-64	T	T	3
58	0.04098	0.05113	0.03053	0.00542	8	128-64	T	T	1
59	0.04098	0.05113	0.03167	0.00626	8	128-64	T	T	2
60	0.04113	0.05118	0.03035	0.00532	8	128-64	T	T	0
61	0.04098	0.05113	0.03044	0.00541	8	128-64	T	F	3
62	0.04098	0.05113	0.03044	0.00541	8	128-64	T	F	1
63	0.04098	0.05113	0.03044	0.00541	8	128-64	T	F	2
64	0.04104	0.05113	0.03043	0.00541	8	128-64	T	F	0
65	0.04098	0.05113	0.03079	0.00577	8	128-64-32	T	T	3
66	0.04098	0.05113	0.03059	0.00558	8	128-64-32	T	T	1
67	0.04098	0.05113	0.03094	0.00572	8	128-64-32	T	T	2
68	0.04098	0.05113	0.03043	0.00540	8	128-64-32	T	T	0
69	0.04098	0.05113	0.03044	0.00541	8	128-64-32	T	F	3
70	0.04098	0.05113	0.03045	0.00541	8	128-64-32	T	F	1
71	0.04098	0.05113	0.03044	0.00541	8	128-64-32	T	F	2
72	0.04098	0.05113	0.03044	0.00541	8	128-64-32	T	F	0
73	0.04098	0.05113	0.03071	0.00561	9	128	T	T	3
74	0.04098	0.05113	0.03057	0.00545	9	128	T	T	1
75	0.04098	0.05113	0.03065	0.00567	9	128	T	T	2
76	0.04096	0.05113	0.03046	0.00543	9	128	T	T	0
77	0.04098	0.05113	0.03045	0.00541	9	128	T	F	3
78	0.04098	0.05113	0.03044	0.00541	9	128	T	F	1
79	0.04098	0.05113	0.03044	0.00541	9	128	T	F	2
80	0.00358	0.00983	0.00639	0.00026	9	128	T	F	0

Table A3: Model 81–120

Index	MSE-train	MSE-val	MSE-test	MSE-WR2	#CL	#Dense	Dropout	BN	L1-L2-Reg
81	0.04098	0.05113	0.03060	0.00561	9	128-64	T	T	3
82	0.04098	0.05113	0.03055	0.00546	9	128-64	T	T	1
83	0.04098	0.05113	0.03112	0.00588	9	128-64	T	T	2
84	0.01292	0.00774	0.00572	0.00423	9	128-64	T	T	0
85	0.04098	0.05113	0.03044	0.00541	9	128-64	T	F	3
86	0.04098	0.05113	0.03044	0.00541	9	128-64	T	F	1
87	0.04098	0.05113	0.03044	0.00541	9	128-64	T	F	2
88	0.04101	0.05114	0.03040	0.00538	9	128-64	T	F	0
89	0.04098	0.05113	0.03073	0.00574	9	128-64-32	T	T	3
90	0.04098	0.05113	0.03053	0.00544	9	128-64-32	T	T	1
91	0.04098	0.05113	0.03100	0.00579	9	128-64-32	T	T	2
92	0.04106	0.05115	0.03040	0.00537	9	128-64-32	T	T	0
93	0.04098	0.05113	0.03044	0.00541	9	128-64-32	T	F	3
94	0.04098	0.05113	0.03045	0.00541	9	128-64-32	T	F	1
95	0.04098	0.05113	0.03044	0.00541	9	128-64-32	T	F	2
96	0.04098	0.05113	0.03044	0.00541	9	128-64-32	T	F	0
97	0.04097	0.05113	0.03235	0.00666	10	128	T	T	3
98	0.04097	0.05113	0.03095	0.00576	10	128	T	T	1
99	0.02726	0.05120	0.03157	0.00643	10	128	T	T	2
100	0.02236	0.01658	0.01053	0.00406	10	128	T	T	0
101	0.04098	0.05113	0.03044	0.00541	10	128	T	F	3
102	0.04098	0.05113	0.03045	0.00541	10	128	T	F	1
103	0.04098	0.05113	0.03044	0.00541	10	128	T	F	2
104	0.04098	0.05113	0.03044	0.00541	10	128	T	F	0
105	0.04098	0.05113	0.03129	0.00604	10	128-64	T	T	3
106	0.04098	0.05113	0.03052	0.00553	10	128-64	T	T	1
107	0.04098	0.05113	0.03089	0.00575	10	128-64	T	T	2
108	0.04112	0.05118	0.03035	0.00532	10	128-64	T	T	0
109	0.04098	0.05113	0.03044	0.00541	10	128-64	T	F	3
110	0.04098	0.05113	0.03044	0.00541	10	128-64	T	F	1
111	0.04098	0.05113	0.03044	0.00541	10	128-64	T	F	2
112	0.04098	0.05113	0.03044	0.00541	10	128-64	T	F	0
113	0.04098	0.05113	0.03066	0.00577	10	128-64-32	T	T	3
114	0.04098	0.05113	0.03080	0.00583	10	128-64-32	T	T	1
115	0.04098	0.05113	0.03159	0.00603	10	128-64-32	T	T	2
116	0.04100	0.05114	0.03041	0.00538	10	128-64-32	T	T	0
117	0.04098	0.05113	0.03044	0.00541	10	128-64-32	T	F	3
118	0.04098	0.05113	0.03044	0.00541	10	128-64-32	T	F	1
119	0.04098	0.05113	0.03044	0.00541	10	128-64-32	T	F	2
120	0.04099	0.05113	0.03044	0.00541	10	128-64-32	T	F	0

Table A4: Model 121–160

Index	MSE-train	MSE-val	MSE-test	MSE-WR2	#CL	#Dense	Dropout	BN	L1-L2-Reg
121	0.04098	0.05113	0.03069	0.00577	11	128	T	T	3
122	0.04098	0.05113	0.03054	0.00552	11	128	T	T	1
123	0.04095	0.05113	0.03123	0.00602	11	128	T	T	2
124	0.04096	0.05113	0.03046	0.00544	11	128	T	T	0
125	0.04098	0.05113	0.03044	0.00541	11	128	T	F	3
126	0.04098	0.05113	0.03044	0.00541	11	128	T	F	1
127	0.04098	0.05113	0.03044	0.00541	11	128	T	F	2
128	0.04100	0.05113	0.03044	0.00541	11	128	T	F	0
129	0.04098	0.05113	0.03088	0.00626	11	128-64	T	T	3
130	0.04098	0.05113	0.03050	0.00546	11	128-64	T	T	1
131	0.04098	0.05113	0.03069	0.00556	11	128-64	T	T	2
132	0.04097	0.05113	0.03045	0.00542	11	128-64	T	T	0
133	0.04098	0.05113	0.03044	0.00541	11	128-64	T	F	3
134	0.04098	0.05113	0.03044	0.00541	11	128-64	T	F	1
135	0.04098	0.05113	0.03044	0.00541	11	128-64	T	F	2
136	0.04100	0.05114	0.03041	0.00538	11	128-64	T	F	0
137	0.04098	0.05113	0.03121	0.00660	11	128-64-32	T	T	3
138	0.04098	0.05113	0.03048	0.00547	11	128-64-32	T	T	1
139	0.04098	0.05113	0.03090	0.00559	11	128-64-32	T	T	2
140	0.04098	0.05113	0.03044	0.00541	11	128-64-32	T	T	0
141	0.04098	0.05113	0.03044	0.00541	11	128-64-32	T	F	3
142	0.04098	0.05113	0.03044	0.00541	11	128-64-32	T	F	1
143	0.04098	0.05113	0.03044	0.00541	11	128-64-32	T	F	2
144	0.04098	0.05113	0.03044	0.00541	11	128-64-32	T	F	0
145	0.04098	0.05113	0.03084	0.00580	12	128	T	T	3
146	0.04098	0.05113	0.03056	0.00558	12	128	T	T	1
147	0.04098	0.05113	0.03068	0.00555	12	128	T	T	2
148	0.00519	0.00107	0.00098	0.00149	12	128	T	T	0
149	0.04098	0.05113	0.03045	0.00541	12	128	T	F	3
150	0.04098	0.05113	0.03044	0.00541	12	128	T	F	1
151	0.04098	0.05113	0.03044	0.00541	12	128	T	F	2
152	0.04098	0.05113	0.03044	0.00541	12	128	T	F	0
153	0.04098	0.05113	0.03086	0.00629	12	128-64	T	T	3
154	0.04098	0.05113	0.03064	0.00554	12	128-64	T	T	1
155	0.04098	0.05113	0.03088	0.00554	12	128-64	T	T	2
156	0.01086	0.00453	0.00340	0.01167	12	128-64	T	T	0
157	0.04098	0.05113	0.03044	0.00541	12	128-64	T	F	3
158	0.04098	0.05113	0.03045	0.00541	12	128-64	T	F	1
159	0.04098	0.05113	0.03044	0.00541	12	128-64	T	F	2
160	0.04098	0.05113	0.03044	0.00541	12	128-64	T	F	0

Table A5: **Model 161–168**

Index	MSE-train	MSE-val	MSE-test	MSE-WR2	#CL	#Dense	Dropout	BN	L1-L2-Reg
161	0.04098	0.05113	0.03084	0.00581	12	128-64-32	T	T	3
162	0.04098	0.05113	0.03056	0.00546	12	128-64-32	T	T	1
163	0.04098	0.05113	0.03098	0.00567	12	128-64-32	T	T	2
164	0.01267	0.01141	0.00600	0.00161	12	128-64-32	T	T	0
165	0.04098	0.05113	0.03044	0.00541	12	128-64-32	T	F	3
166	0.04098	0.05113	0.03044	0.00541	12	128-64-32	T	F	1
167	0.04098	0.05113	0.03044	0.00541	12	128-64-32	T	F	2
168	0.04098	0.05113	0.03044	0.00541	12	128-64-32	T	F	0