

Structure Prediction from Neutron Scattering Profiles: A Data Sciences Approach

Cristina Garcia-Cardona^a, Ramakrishnan Kannan^b, Travis Johnston^b, Thomas Proffen^c and Sudip K. Seal^b

^aComputer, Computational and Statistical Sciences Division, Los Alamos National Laboratory, USA

^bComputer Science and Mathematics Division, Oak Ridge National Laboratory, USA

^cSpallation Neutron Source, Oak Ridge National Laboratory, USA

Abstract—One of the main goals of neutron data analysis is to determine the internal structure of materials from their neutron scattering profiles. These structures are defined by a crystallographic class label and a set of real-valued parameters specific to that class. Existing structure analysis approaches use computationally expensive loop refinements methods that routinely take days, and even weeks, to complete. Additionally, the outcomes often rely on the fidelity of physical models that are computed during the refinement process.

Here, we evaluate the feasibility of using trained data-driven machine learning models as fast and accurate substitutes for these expensive methods. We report on the efficacies of a variety of ML models, including convolutional neural networks, auto-encoders, random forests and combinations thereof, in addition to techniques such as transfer learning in predicting these structural parameters. Specifically, we evaluate two categories of models which we call class-conditional and integrated. The first relies on a two-stage inference pipeline in which a crystallographic class label is first predicted followed by regression to predict the length/angle parameters. In the second category, the classification and regression tasks are performed as a single learning task. We train these models on synthetically generated data, validate them against experimental observations and show that integrated models outperform their class-conditional counterparts opening up the possibility of deep learning models as a viable alternative to existing resource-intensive loop refinement methods in neutron data analysis.

I. INTRODUCTION

The ability to design customized material with targeted mechanical and chemical properties relies on a detailed understanding of their internal structure. Neutron scattering is a state-of-the-art experimental technique that allows scientists to probe material structures with atomic resolutions by scattering beams of neutrons from them. The scattered

This manuscript has been authored by UT-Battelle, LLC under Contract No. DE-AC05-00OR22725 with the U.S. Department of Energy. The United States Government retains and the publisher, by accepting the article for publication, acknowledges that the United States Government retains a non-exclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for United States Government purposes. The Department of Energy will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (<http://energy.gov/downloads/doe-public-access-plan>) A portion of this research used resources at the Spallation Neutron Source, a DOE Office of Science User Facility operated by the Oak Ridge National Laboratory. A portion of this research used resources at the Argonne Leadership Computing Facility, a DOE Office of Science User Facility operated by the Argonne National Laboratory. This research was sponsored by ExaLearn, an Exascale Computing Project, DOE.

neutrons are collected using different types of detectors and represented in a two-dimensional scattering intensity plot called a *Bragg profile*. One of the major tasks in neutron data analysis is to solve the inverse problem of determining the internal structure of the target material based on its observed Bragg profile.

A. Motivation

Existing methodologies to solve the aforementioned inverse problem largely rely on loop refinement techniques wherein a forward physics-based model uses an initial guess for the internal structure to generate a scattering Bragg profile which is then compared with the observed pattern. If the patterns are found to match, based on a pre-defined definition of similarity, the initial guess is assumed to accurately mirror the internal structure of the material. Else, the guess is modified and the entire process repeated. In practice, this loop refinement technique is very time-consuming as the number of iterations required to converge to an acceptable level of similarity between the computed and observed Bragg profiles vary widely between samples. More often than not, it requires hundreds and thousands of iterations to converge and routinely takes many days, even weeks, to complete. The overall time-to-solution and the quality of results from loop refinement methods also depend on the fidelity of the forward model used to generate the Bragg profiles within the loop iterations. Higher fidelity usually translates to better quality of solutions but requires even longer time-to-solution. On the other hand, shorter time-to-solutions with low-fidelity forward models compromise quality of solutions. The main motivation of the work presented here is to evaluate alternative data-driven approaches to accelerate this discovery process while circumventing these trade-offs inherent in loop refinement methods.

B. Definitions

Crystalline materials belong to seven crystallographic classes and 14 Bravais lattices [1] as shown in Fig. 1. Each Bravais lattice is characterized by a set of unit cell lengths, denoted by the parameter set $\{a, b, c\}$, and unit cell angles, denoted by the parameter set $\{\alpha, \beta, \gamma\}$. Depending on the crystallographic class of the material, these parameters

satisfy unique constraint relations, as shown in Fig. 1. Therefore, given the Bragg profile of a material sample, a trained ML model needs to predict the crystallographic class to which it belongs as well as the unit cell lengths and angles that satisfy the relations conditioned on that class.

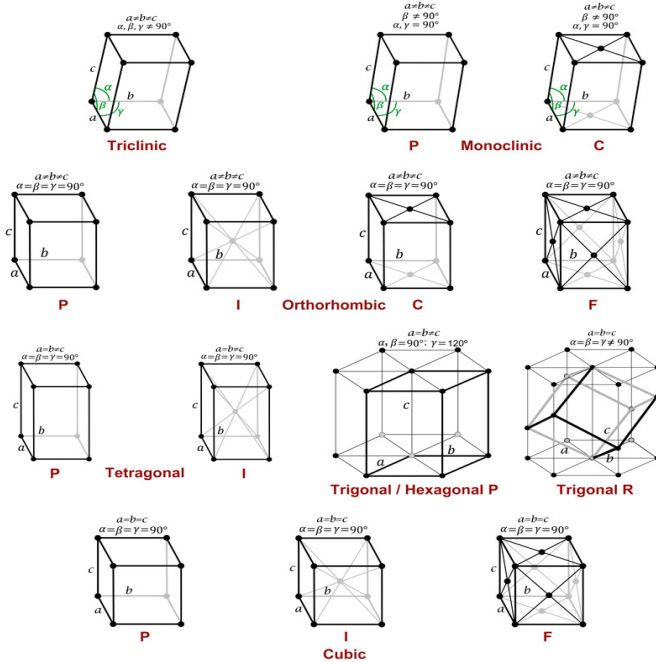


Figure 1: The 14 Bravais lattices (7 crystal classes) compatible with three-dimensional translational periodicity. [2]

C. Challenges

This paper evaluates the efficacy of replacing loop refinement methods with pre-trained machine learning (ML) models as inference engines for predictions of the structure parameters of material samples based on the information embedded in their Bragg profiles. However, data-driven ML approaches for this problem encounters a host of challenges. One of the major challenges in adopting an ML approach is the paucity of labeled neutron data with which to train ML models. Since neutron experiments are extremely resource-intensive and cannot be carried out on-demand, the amount of observed experimental data (Bragg profiles) that are reliably labeled is very limited. On the other hand, accurately training ML models requires large amounts of training data. Another challenge is that the number and kinds of parameters to be predicted differ from one material to another depending on its crystallographic class. Thus, a ML model should not only be able to predict the class of the material but also the correct set of cell length/angle parameters that correspond to that class. Additionally, experimental observations collected from neutron detectors are mixed in with background noise that are unique to each

detector experiment. Identifying the signal from the noise and using this information to wrangle the data for more accurate model training poses yet another challenge. Finally, the sampling space spanned by the structural parameters $\{a, b, c\}$ and $\{\alpha, \beta, \gamma\}$ grows exponentially with the number of parameters that define each symmetry class.

D. Related Work

ML-driven methods for structure determination from neutron scattering data is an emerging area of research. Recently, auto-encoders have been demonstrated to be effective in extracting spin Hamiltonians from neutron scattering data [3]. Principal component analysis with an artificial neural network was shown to predict neutron scattering cross-sections to constrain the parameters of a pre-existing model Hamiltonian in [4]. An unsupervised ML approach to study phase transitions in single crystal x-ray diffraction data was reported in [5]. An ML-based approach to classify the local chemical environment of specific metal families from the simulated K-edge XANES of a large number of compounds was reported in [6]. The use of ML in understanding neutron physics is beginning to gain greater acceptance as highlighted recently in [7]. To our knowledge, the results reported here represent one of the first efforts in this direction and makes multiple advances beyond the findings based on shallow ML models presented by the authors in [8].

E. Contributions

The work presented here makes multiple advances in the application of ML models to scientific knowledge discovery in the neutron sciences. Specific contributions are the following:

- We demonstrate that deep learning models perform significantly better than shallow learning models [8] in predicting structure parameters from neutron scattering data.
- We show that transfer learning techniques can be gainfully leveraged to build unified deep learning models that predict both the class labels and the class-dependent parameters from Bragg profiles with acceptable levels of accuracy.
- We present new heuristic methodologies to control the effects of background noise in the learning task.
- We conclusively demonstrate that an integrated model that predicts the class as well as the class-specific parameters in a single learning task performs better than class-conditional models which learn to predict the class and the class-dependent parameters as separate learning tasks.

Use of ML models for structure prediction from neutron scattering data is a very nascent field and, to the best of our knowledge, the methodology and results presented here have not been reported before. The rest of the paper is organized

as follows. Section II provides a brief overview of the type of machine learning models used in this study. Section III describes the data generation methods and specifications. In Section IV, the class-conditional and integrated models are described followed by the results of our experiments in Section V. We draw some conclusion in Section VI.

II. PRELIMINARIES

In this section, the network models used in the remainder of the paper are briefly described.

A. Random Forest

Random forest (RF) is a type of ensemble predictor that aggregates results of distinct decision trees to solve classification or regression problems [9]. The aggregation of individual results improves the performance of the model reducing the variance in the predictions and leading to good generalization over data not used for training. More about RFs and their implementation can be found in [8]–[11].

B. Convolutional Neural Network:

A convolutional neural network (CNN) [12] is a feed-forward neural network composed of convolution and pooling layers. Each node n in a convolutional layer i computes the following operation:

$$y_n^i = \sigma_{i,n} \left(\sum_m^{M^i} \mathbf{w}_{n,m}^i * \mathbf{x}_m^i + b_n^i \right)$$

where \mathbf{x}_m^i represents the m -th input map at layer i ; $\mathbf{w}_{n,m}^i$, b_n^i and $\sigma_{i,n}$ represent node parameters, namely, the m -th convolutional kernel, the bias and the activation function, respectively; M^i are the number of input maps at layer i (which correspond to the output maps at layer $i-1$, i.e. the number of nodes at layer $i-1$); and $*$ denotes a convolutional operation. The activation function is usually a rectified linear unit (ReLU) which corresponds to the following operation $\text{ReLU}(\nu) = \max(0, \nu)$. The pooling layer i reduces the dimensionality of the input pattern by averaging (or taking the maximum) over a fixed neighborhood structure in the input pattern, while sweeping the pattern with a consistent stride, and making the model more robust to local variations in the input. Note that the size of the stride used determines the extent of dimensionality reduction, e.g. if the pooling uses a stride of 2 the dimensionality is reduced to half and so on.

As any other neural network model, the CNN network is trained to produce a desired output by minimizing a loss function over the training data set. The minimization determines a set of model parameters, corresponding to the values of the convolutional kernels and the biases, that better approximate the desired output. However, since the node operations are based on convolutions that take into account the entire signal, and since the convolutional

kernels are compactly supported kernels of low dimensionality (typically 3-11 components in 1D), CNNs are able to capture local invariant patterns in the data, which are optimal throughout the input signal. This also implies that the number of parameters is much less than it would be required if the layer was a regular (i.e. densely) connected layer, thereby yielding a more compact representation. In summary, CNN models capture local correlations using a much lower number of parameters than regular densely connected neural networks.

C. Auto-Encoder

An auto-encoder consist of two parts: an encoder and a decoder [13], [14]. The encoder maps the input into a hidden (latent) representation. The decoder is able to use the hidden representation and map it back to the input space. Auto-encoders are used for many applications. One possible application is dimensionality reduction: the encoder maps the input pattern to a latent representation with a smaller dimensionality than the input space, while the decoder tries to minimize the error in reconstructing the original pattern when mapping the latent representation back to the input space. Thus, an auto-encoder is a type of unsupervised model that can be trained by minimizing the error between input and output patterns. Since the error of reconstruction from a latent representation is minimized, it is deemed that the latent representation captures the essential features of the pattern, and since the latent representation has a smaller dimensionality than the input space, the auto-encoder trained in this way can be regarded as a dimensionality-reduction model. A convolutional auto-encoder (CAE) uses CNNs to build the encoder, the decoder, or both [15].

III. DATA GENERATION

To address the challenge of scarcity of labeled data, the six-dimensional parameter space, collectively denoted by Y , where Y represents the set $\{a, b, c, \alpha, \beta, \gamma\}$ of unit cell parameters, is uniformly sampled (within appropriate ranges established from domain knowledge) and a Bragg profile is computed at each sampled point using the Generalized Structure Analysis System (GSAS), a widely used structure refinement software in the neutron and X-ray crystallography community [16]. GSAS-II requires two sets of input specifications to simulate a diffraction pattern: the crystallographic class information and the instrument description. The first allows determination of the appropriate physics-driven constraint equations corresponding to the symmetry class, while the second allows modelling of the diffractometer physics used to generate the diffraction profile.

In this preliminary work, we limit the scope of our study to a perovskite material called *barium titanate* ($BaTiO_3$). Since barium titanate, without doping, exists only in three of the fourteen possible lattice groups, labeled training data sets are generated only for the tetragonal, trigonal and cubic

crystallographic symmetry classes using GSAS-II. The instrument specification used was for the NOMAD instrument [17] to maintain consistency with the NOMAD-generated experimental data against which the model predictions are subsequently validated.

Diffraction patterns from barium titanate in cubic, trigonal and tetragonal crystallographic classes were generated to build the labeled training set. For cubic class, a was sampled in the range [3.5, 4.5] with step 10^{-3} . For the trigonal class, a was sampled in the range [3.8, 4.2] with step 10^{-3} , while α in the ranges $[60^\circ, 89.8^\circ]$ and $[90.5^\circ, 120^\circ]$ with step 0.5° . For the tetragonal class, a, c were sampled in the range [3.8, 4.2] with step 10^{-3} . We used a time-of-flight (ToF), in the range $[1,360\mu\text{s}, 18,919\mu\text{s}]$ with step $0.0009381\mu\text{s}$. Sweeping over these ranges yields a collection of diffraction patterns. Each diffraction pattern X is a set of 2807 2-tuples $(x, I(x))$ where x is the time-of-flight (ToF) and $I(x)$ is the GSAS-generated scattering profile. Table I lists the relations for the three symmetry classes used in this study.

Table I: Training Data Set of Labeled Neutron Diffractions.

Class	Parameters	Samples (n)	Size
Cubic (predict a)	$a = b = c$	1,000	43 MB
Trigonal (predict a, α)	$a = b = c$ $\alpha = \beta = \gamma \neq 90^\circ$	47,719	2 GB
Tetragonal (predict a, c)	$a = b \neq c$ $\alpha = \beta = \gamma = 90^\circ$	160,400	6.8 GB

Depending on the particular combination of crystallographic symmetry and structural parameters, each labeled sample in the training sets required between 2s to 30s to compute. An MPI-based parallel framework [8] was developed to generate the diffraction patterns in a concurrent and distributed manner.

IV. MODELS

Recall that the cumulative task of predicting the structural parameters consists of predicting a class label as well as the unit cell parameters corresponding to that class. The first is a classification task while the second a regression task. As mentioned in Section I, the number of cell parameters to be predicted varies with the predicted class. To address this challenge of conditional predictions, two categories of models were tested and their accuracies compared. In the first category, called *class-conditional models* (denoted by **C**), the overall prediction is carried out in a sequence of two independent learning tasks. In the first task, a classifier predicts the crystallographic symmetry and, in the second, a regressor predicts the cell lengths/angles. The second category, called *integrated or multi-task models* (denoted by **I**), are designed to predict the symmetry class and the cell lengths/angles in a single ML task. Central to both categories of models is the classifier which predicts the crystallographic class. This is described next.

A. Classifier for Crystallographic Symmetry

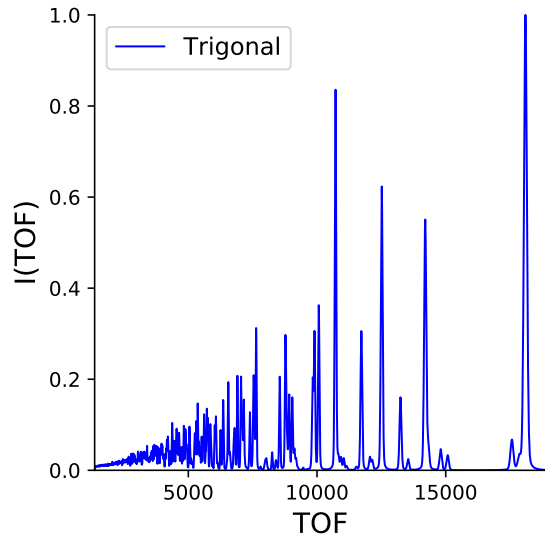


Figure 2: Example of Intensity vs Time of Flight curves (Bragg profile) used as training data.

A 1D convolutional neural network was trained on data generated by GSAS-II (described in Section III) to distinguish training samples belonging to the tetragonal, trigonal and cubic classes. Specifically, the CNN was designed to accept an input vector of length 2,807 representing the normalized intensities of each curve and predict (classify) the crystallographic symmetry group it belongs to. An example of an intensity vs. time of flight (ToF) curve (Bragg profile) is shown in Fig. 2. The neural network structure that was trained is as follows:

- Feature Learner
 - 1D Convolution (16 learned filters, kernel width=3, stride=1)
 - 1D Max Pooling (kernel width=2, stride=2)
 - 1D Convolution (32 learned filters, kernel width=4, stride=2)
 - 1D Max Pooling (kernel width=2, stride=2)
 - Fully Connected (256 hidden neurons, w/ReLU activation)
- Fully Connected (3 output neurons, w/Softmax)

Training used stochastic gradient descent with a fixed learning rate of 0.001, weight decay of .005, momentum of 0.9, and a batch size of 90. Since the number of examples in each of the three groups is not balanced (128K Tetragonal examples, 38K Trigonal examples, and only 800 cubic examples) each mini-batch was constructed by randomly sampling an equal number of examples from each class, i.e. 30 examples from each class. The training and validation losses during training is shown in Figure 3.

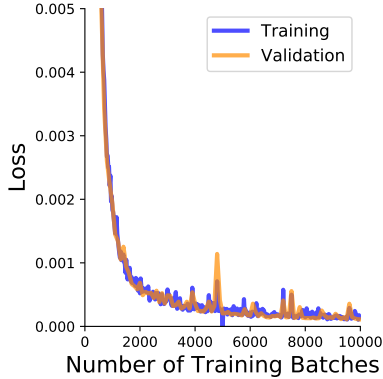


Figure 3: Training and validation losses of the classifier.

B. Class-Conditional Models

1) *Class-Conditional Random Forest, C₁*: Here three different class-conditional RF models were trained. Each model was trained independently to predict the unit cell parameters for cubic, tetragonal and trigonal symmetries, respectively. Hence, given an input corresponding to a diffraction pattern and given the corresponding symmetry classification, the appropriate regression model is trained and used afterwards for prediction. The cubic model is trained to predict a , the tetragonal model is trained to predict a and c and the trigonal model is trained to predict a and α . In order to control the complexity of the RF and to prevent the models from overfitting, we limited the depth of the individual trees and set a maximum number of individual trees in the forest.

2) *Transfer Learner, C₂*: It is clear from Fig. 3 that the fine-tuned classifier discussed in Section IV-A classifies the cubic, tetragonal and trigonal classes with high accuracy in the synthetic data set. The class-conditional model discussed here, called the *transfer learner*, leverages the features (learnt with high accuracy) by the classifier to regress the continuous parameters Y . Note that for the tetragonal class, $Y = \{a, c, 90^\circ\}$; for the trigonal class, $Y = \{a, a, \alpha\}$; and for cubic, $Y = \{a, a, 90^\circ\}$. For each Bragg profile (training sample), the trained classifier from Section IV-A is run and 256 features learned by the first fully connected layer is used as the input features of a fully connected regressor model to estimate the continuous parameter set Y updating only the weights of this fully connected during back propagation. Note that the regressor still trains separately conditioned on the predicted class.

3) *Deep Regressor, C₃*: As a natural extension to the transfer learner model, a second class-conditional model is evaluated in which the last layer is replaced with a linear layer to produce continuous outputs Y using mean square error to compute the loss and update the weights of *all* the layers during back-propagation and not just the weight of the linear regressor. Note that this too is a class-conditional model, which we refer to as *deep regressor*. In this model,

the weights from the classifier are only used to initialize the corresponding layers in Deep Regressor.

C. Integrated Models

Recall that integrated models predict both the class labels and the regression values of the unit cell lengths/angles in a single prediction task. Accordingly, the integrated models predict four outputs, namely, the class label, S (0: cubic, 1: tetragonal and 2: trigonal), the lattice parameter a , the lattice parameter c and the angle parameter α . Note that these four predictions form the minimal set of parameters necessary to determine the structures of the three crystal symmetries studied here (see Table I).

1) *Deep MultiTask, I₁*: This multitask network, referred to as *Deep MultiTask*, uses the output from the first fully connected layer of the classifier from Section IV-A to train both a regressor (using softmax) in addition to the original classification task. The classifier updates the weights using the error obtained from cross entropy loss while the regressor uses MSE loss for every batch.

2) *Random Forest, I₂*: Instead of building a class-conditional regression model for each of the symmetries, random forests were used to train an integrated model. In this case a unique regression model with four outputs is trained with the goal of predicting the class of the crystal symmetry and the unit cell parameters (a , c and α) for a given input corresponding to a Bragg profile.

3) *CAENN, I₃*: In order to construct features sensitive to local correlations in Bragg profiles, a deep learning model based on CNN was implemented. Specifically, a 1D CNN auto-encoder (CAE) was combined with a neural network (NN) regressor. The CAE helps to find a good latent representation for a given Bragg profile. This latent representation is used as the input to an integrated NN-based regressor that predicts the crystal symmetry and the unit cell parameters. Since all the Bragg profiles share the same x component (ToF), each profile, though two-dimensional, can be regarded as a one-dimensional pattern with the understanding that the first dimension is common to all the samples. As such, a 1D symmetrical bottle-neck CAE suffices. The CAE is designed with a symmetrical bottle-neck architecture of 4 layers (total), the thin part of which corresponds to a latent representation. Note that some symmetrical bottle-neck auto-encoders are built with tied weights, but in our case, we learned different (not-tied) weights for encoder and decoder. Finally, an integrated NN regressor with 4 densely connected layers and ReLU activations is built to predict simultaneously the crystal symmetry and unit cell parameters (a , c and α) for a given Bragg profile.

V. RESULTS

The models described above were trained using data generated by GSAS-II, as described in Section III. These trained models were then used as inference engines to predict

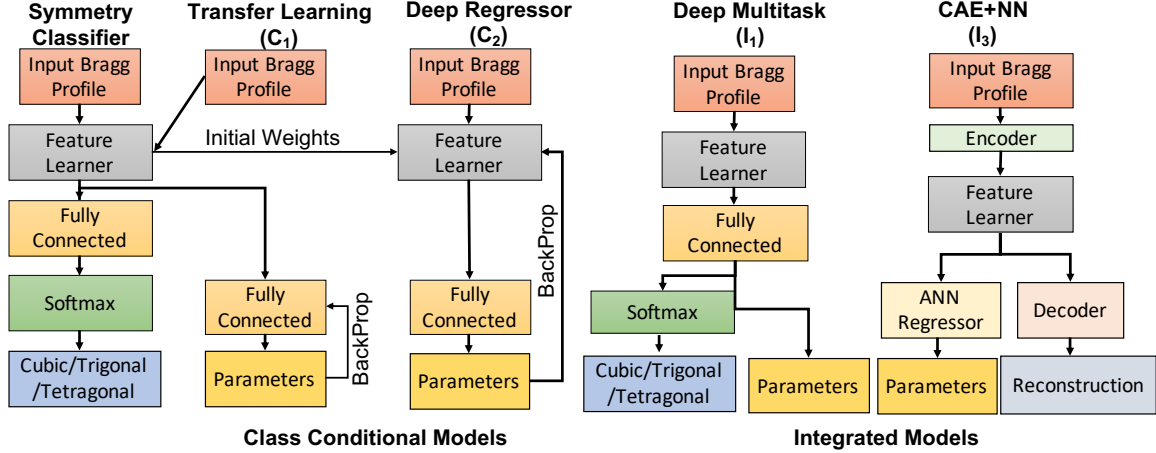


Figure 4: Classifier, class-conditional models (C_2 and C_3) and integrated models (I_1 and I_3).

the structure parameters using experimental Bragg profiles collected using the NOMAD diffractometer for barium titanate samples. The predicted parameters were then used as inputs to the GSAS-II simulator and the resulting simulated Bragg profile was compared with the experimental Bragg profile for validation. The remainder of this section presents the results of these experiments using models described in the previous section.

A. Metrics

Since we evaluate the performance of class conditional and integrated models on the parameters a , c and α , we use Mean Squared Error (MSE).

$$\text{MSE}(Y, \hat{Y}) = \frac{1}{n} \sum_{i=1}^3 \|Y_i - \hat{Y}_i\|_2^2, \quad (1)$$

The vectors $Y_i, \hat{Y}_i \in \mathbb{R}^n$ represent the ground truth and predicted value of the parameters a, c and α , respectively. The mean squared error reported on class conditional models and integrated models are different with respect to the number of samples n . In the case of class conditional models, $Y \in \mathbb{R}^{n \times 3}$, where n is the total test samples for each of the class as detailed in Section III. For an integrated model, however, $n = 41,824$ includes samples from all three symmetric classes, viz., cubic, trigonal and tetragonal.

B. Synthetic Data

The simulated diffraction patterns were split into two sets: 80% for training and 20% for testing. This corresponds to the following sizes:

- Cubic: 800 diffraction patterns for training and 200 for testing.
- Trigonal: 38,175 diffraction patterns for training and 9,544 for testing.
- Tetragonal: 128,320 diffraction patterns for training and 32,080 for testing.

- Integrated: 167,295 diffraction patterns for training and 41,824 for testing.

All the data was pre-processed in the same way. Specifically, each histogram was vertically shifted to the minimum value of 0 and then scaled appropriately to the maximum value of 1. The following subsections describe the different models that were trained and the results of evaluating them on the test sets (samples not seen during training).

1) *Random Forest (RF)*: The following parameters were used to train the RF models. Cubic: a RF of 150 trees with maximum depth of 50, trigonal: a RF of 200 trees with maximum depth of 50 and tetragonal: a RF of 200 trees with maximum depth of 50. The integrated model corresponds to a RF of 250 trees with maximum depth of 50.

For training a random subsample over the training set was used (cubic: 792; trigonal: 6,500; tetragonal: 20,500 and integrated: 27,800 diffraction patterns). Note that to reduce the noise, the left-most and the right-most ends of the diffraction patterns were discarded, specifically 256 and 55 pairs of $(x, I(x))$ values at the low and the high TOF (x) ends, respectively, resulting in training patterns of dimensionality 2,496. This heuristic is an ad hoc attempt motivated by the experimental patterns available, which exhibit rapid oscillations at the low TOF end and a saturated zero-signal at the high TOF end. Evaluation of other strategies to reduce noise at pre-processing stages will be part of future work.

2) *Transfer Learning*: Note that the two class-conditional models, C_2 and C_3 , and the integrated model, I_1 , are all variants of transfer learning approaches. These three models were trained using a batch size of 512 and trained for 500 epochs. In order to produce balanced classes, 800 cubic samples were used with replacement in every epoch. We used ADAM optimizer with learning of 10^{-3} for backpropagation.

3) *CAENN*: The following parameters were used to train the CAENN model. Note that the tails of the Bragg profiles are discarded yielding a 2,496 dimensionality for the input

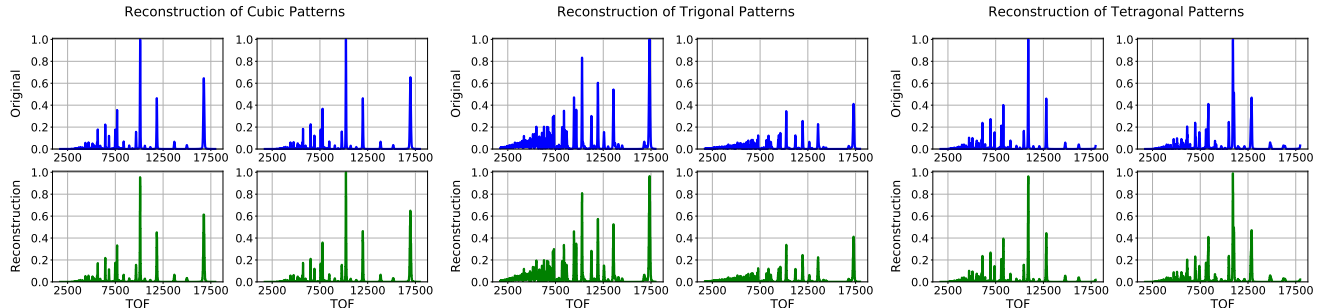


Figure 5: Examples of CAE reconstruction for test samples. For each class, two random samples were reconstructed.

patterns like in the RF model.

CNN Auto-Encoder: The 1D CAE is designed with a symmetrical bottle-neck architecture of 4 layers total: 2 for the encoder and 2 for the decoder. The structure that was trained for the encoder is the following:

- 1D Convolution (9 learned filters, kernel width=15, stride=1) followed by ReLU activation.
- 1D Max Pooling (kernel width=2, stride=2)
- 1D Convolution (2 learned filters, kernel width=7, stride=1) followed by ReLU activation.
- 1D Max Pooling (kernel width=2, stride=2)

The decoder is a symmetrical reflection of this structure, with weights that are also learned during CAE training.

A total of 34,100 diffraction patterns from the three different symmetries were used for training. This is about 20% of the tetragonal and trigonal training sets and 800 diffraction patterns of the cubic symmetry. The CAE was trained for 250 epochs, to minimize the mean absolute error function, using an Adaptive Moment Estimation (ADAM) [18] optimizer with a learning rate of 10^{-3} and a batch size of 20. The best model over 5-fold cross validation evaluated over the testing set is selected as the CAE model. The testing MSE for this CAE is 85.34. Randomly selected examples of reconstructions for samples in the test set for each of the three crystal symmetries cubic, trigonal and tetragonal evaluated using the best CAE model are shown in Figure 5.

NN Regressor: The integrated NN regressor has a four-layer structure with 150, 70 and 20 nodes in the intermediate layers and 4 outputs. The random subsample of 34,100 diffraction patterns that was used for training the CAE is also used to train the NN. The regressor was trained to minimize an ℓ_2 -regularized MSE function with 10^{-4} regularization weight, during 500 epochs, using an ADAM optimizer with an initial learning rate of 10^{-3} and a decaying factor of 0.1 on epochs 100, 250 and 400, and a batch size of 20.

4) *Model Performance:* Table II and Table III summarize the performance of the different models on the synthetic dataset. Table II shows that class conditional RF performs better in the case of cubic class which requires prediction of only one parameter a . However, when multiple labels need to be predicted, as in trigonal and tetragonal classes,

the class conditional Deep Regressor model performs better. The CAENN integrated models outperformed both the RF and Multitask networks. The CAENN and Multitask models were trained with 500 epochs each. A longer training of the models has the potential to further improve the results.

Table II: MSE: Class Conditional Models – Synthetic Test Set. Transfer learning based models C_2 and C_3 outperform RF for multilabel scenarios.

Symmetry	Model MSE		
	RF (C_1)	Transfer Learning (C_2)	Deep Regressor (C_3)
Cubic	1.20×10^{-6}	4.71×10^{-4}	1.00×10^{-5}
Trigonal	8.14×10^{-4}	1.01×10^{-2}	1.40×10^{-5}
Tetragonal	5.84×10^{-6}	5.18×10^{-4}	2.60×10^{-5}

Table III: MSE: Integrated Models – Synthetic Test Set. Integrated models perform better than class conditional models

Model	MSE
RF (I_1)	1.48×10^{-4}
Multitask(I_3)	1.90×10^{-5}
CAENN (I_2)	5.96×10^{-6}

C. Experimental Data

Experimental neutron powder diffraction data of barium titanate as a function of temperature was collected on the NOMAD instrument housed in the Spallation Neutron Source at Oak Ridge National Laboratory. In all cases, "traditional" structure analysis was carried out to obtain the structural parameters. The crystallographic class and the lattice parameter set $\{a, c, \alpha\}$ were used as labels and the machine learning models were evaluated against this ground truth set. A total of 15 experimental diffraction patterns were evaluated – one belonging to the trigonal symmetry class and the remaining fourteen belonging to the tetragonal class.

1) *Data Pre-processing:* As mentioned in Section I, experimental signals contain detector-specific background noise. Background signals in neutron detectors originate from a variety of sources (diffuse scattering, air scattering, detector readout noise and others) and need to be subtracted out to improve the signal-to-noise ratio. A second-order

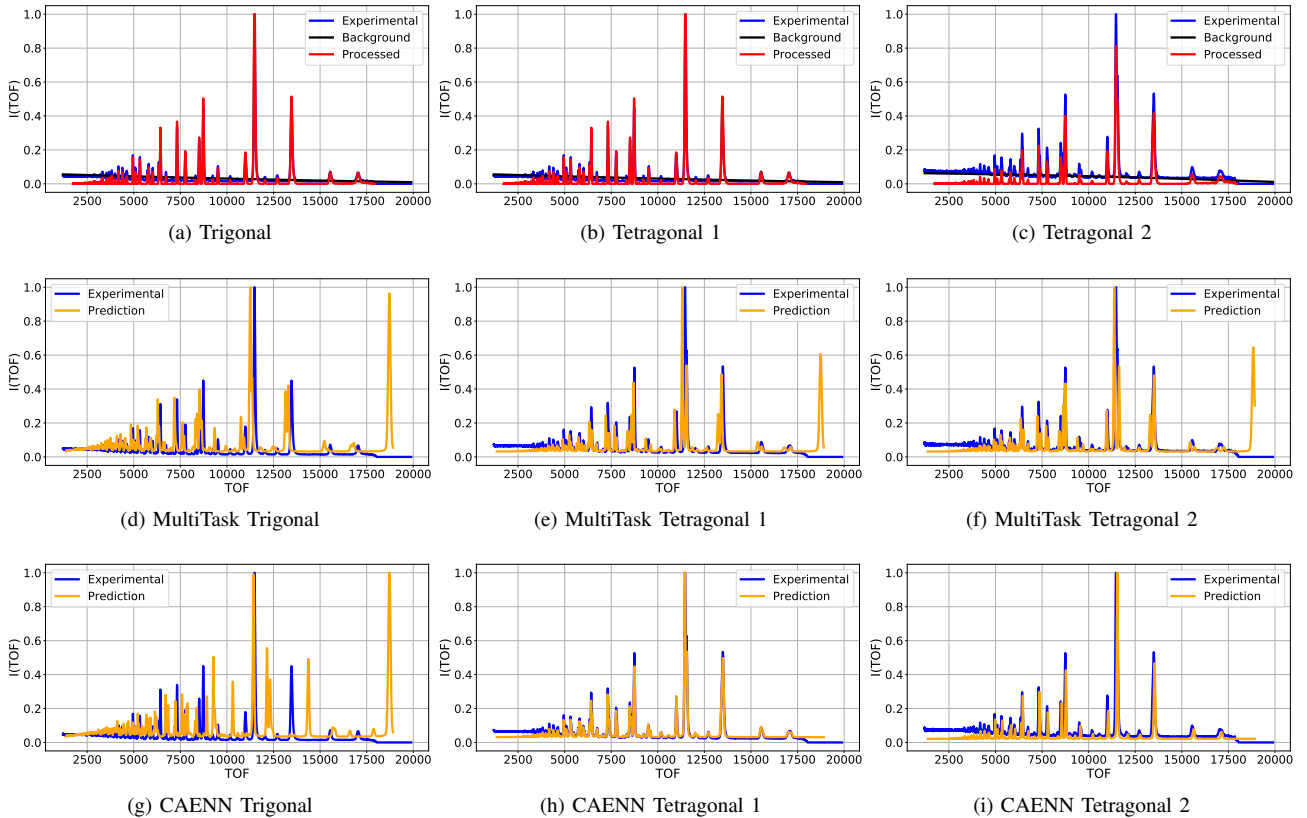


Figure 6: Examples of Integrated model predictions for experimental data. Comparison of experimental measurements and diffraction patterns generated from predicted unit cell parameters using Multitask and CAENN. Top: pre-processing of the first three experimental patterns. The ground truths for trigonal are $a = 3.9968$ and $\alpha = 89.84^\circ$, Tetragonal 1 are $a = 3.9857$ and $c = 4.0277$, and for Tetragonal 2 $a = 3.9870$ and $c = 4.0279$. Center left: Multitask (MT) trigonal prediction for $a = 3.9272$ and $\alpha = 89.3611$. MT tetragonal 1 predicted $a = 3.9318, c = 4.0414$ and MT tetragonal 2 predicted $a = 3.9510, c = 4.0752$. Bottom left: trigonal, prediction $a = 4.0094$ and $\alpha = 97.61^\circ$. Bottom center: tetragonal prediction $a = 3.9851$ and $c = 4.0358$. Bottom right: tetragonal prediction $a = 4.0196$ and $c = 4.0210$.

Chebyshev polynomial of the first kind is used to model a NOMAD-specific background signal for each experimentally observed diffraction pattern independently. A signal threshold in the experimental Bragg profile is adjusted such that the area under the profile closely matches (differs by less than 10^{-4}) with that under the Chebyshev polynomial. This polynomial is then subtracted out from the original experimental signal. Preliminary results indicate that this method is more robust to experimental conditions than previous quantitative measures used. In addition to the background corrections, the x -axis (ToF) also needs to be adjusted for better consistency between the simulated and experimentally collected Bragg profiles. For this, each $(x, I(x))$ pair in the experimental diffraction pattern is matched with the closest TOF from the simulated ToF (which is the same for all the GSAS-II generated diffraction patterns). Finally, the intensities, $I(x)$, for the experimental and Bragg profiles are clipped to the $[0,1]$ range. Examples of the pre-processing of

experimental data, including the background estimation and the processed experimental diffraction pattern, are included in Figure 6 (top row) for the three first experimental data samples.

2) *Model Performance*: Table IV and Table V summarize the performance on the experimental data. Figure 6 (center and bottom rows) shows the comparison of the three first experimental data samples with the diffraction patterns generated from the predicted unit cell parameters.

Table IV: MSE: Class Conditional – Experimental Data.

Symmetry	Model MSE		
	RF (C_1)	Transfer (C_2)	Deep (C_3)
Trigonal	6.30×10^{-2}	1.91×10^{-1}	6.75×10^{-2}
Tetragonal	1.08×10^{-2}	3.08×10^{-2}	3.08×10^{-4}

The main outcome of these experiments is that optimizing the combination of cross entropy and MSE losses is more effective than using integer class labels and optimizing

Table V: MSE: Integrated Models – Experimental. Deep learning based models predict experimental parameters with more accuracy than RF

Model	MSE
RF (I_1)	1.17×10^{-1}
Multitask (I_2)	1.23×10^{-3}
CAENN (I_3)	2.00×10^{-3}

the MSE loss. This is mainly due to the fact that the minimization of the cross entropy effectively maximizes the likelihood of the classes predicted by the model, which is not guaranteed for the minimization of the MSE loss over integer labels.

VI. CONCLUSIONS

This paper demonstrates the viability of a data-driven approach to the long-standing problem of determining material structure from neutron scattering data. Existing methods are extremely time-consuming and rely on the fidelity of physics-driven forward models for accuracy. The alternative presented here is fast, data-driven and less reliant on the fidelity of the underlying physics. Using perovskite as an example material sample, the paper reports an extensive comparative study of the efficacies of multiple deep learning ML models (and combinations thereof) in predicting its structural parameters. The overall structure prediction task involves the classification task of predicting the crystallographic class that the sample belongs to and the regression task of predicting the unit cell lengths/angles corresponding to that class.

In this context, we presented the performance results of two types of ML models – class-conditional and integrated. Class-conditional models learn the class followed by the corresponding cell parameters in a sequence of two learning tasks while the integrated models learn them both in a single learning task. Multiple variants of these two broad categories were trained using synthetically generated data. These trained models were validated against experimental data and good prediction accuracies were obtained. Overall, we note that deep learning models benefit more from the multi-task approach than RF models. For the integrated models, we found that optimizing the combination of cross entropy and MSE losses is more effective than using integer class labels and optimizing the MSE loss, mainly due to the fact that minimization of the cross entropy effectively maximizes the likelihood of the classes predicted by the model, which is not guaranteed for the minimization of the MSE loss over integer labels. In future, multi-task networks that can predict the crystallographic classes not studied in this report will be trained in addition to exploring more sophisticated transfer learning models capable of accurate predictions on experimental diffraction patterns gathered from a wider range of diffractometers.

REFERENCES

- [1] C. Kittel, *Introduction to Solid State Physics*, 8th ed. Wiley, 2004.
- [2] M. Martinez-Ripoll. (2014) Crystallography. Departamento de Cristalografía y Biología Estructural. [Online]. Available: <http://www.xtal.iqfr.csic.es/Cristalografia/>
- [3] A. M. Samarakoon, K. Barros, Y. W. Li, M. Eisenbach, Q. Zhang, F. Ye, V. Sharma, Z. L. Dun, H. Zhou, S. A. Grigera, C. D. Batista, and D. A. Tennant, “Machine-learning-assisted insight into spin ice $\text{dy}_2\text{ti}_2\text{o}_7$,” *Nature Communications*, vol. 11, no. 1, p. 892, 2020.
- [4] R. Twyman, S. Gibson, J. Molony, and J. Quintanilla, “A machine learning approach to magnetic neutron scattering,” March 2019.
- [5] J. Venderley, M. Matty, and E.-A. Kim, “Unsupervised machine learning of single crystal x-ray diffraction data,” March 2019.
- [6] D. Lu, M. Carbone, M. Topsakal, and S. Yoo, “Using machine learning to predict local chemical environments from x-ray absorption spectra,” March 2019.
- [7] S. J. Tony Hey, Keith Butler and J. Thiyagalingam, “Machine learning and big scientific data,” *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 378, no. 2166, 2020.
- [8] C. Garcia-Cardona, R. Kannan, T. Johnston, T. Proffen, K. Page, and S. K. Seal, “Learning to predict material structure from neutron scattering data,” in *2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, December 9-12, 2019*, 2019, pp. 4490–4497.
- [9] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, Oct. 2001. [Online]. Available: <https://doi.org/10.1023/A:1010933404324>
- [10] A. Géron, *Hands-On Machine Learning with Scikit-Learn & TensorFlow*, 1st ed. O’Reilly, 2017.
- [11] “scikit-learn: Machine Learning in Python,” 2020. [Online]. Available: <https://scikit-learn.org/stable/>
- [12] Y. LeCun, Y. Bengio, and G. E. Hinton, “Deep learning,” *Nature*, vol. 521, pp. 436–444, 2015.
- [13] P. Vincent and H. Larochelle, “Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion,” *Journal of Machine Learning Research*, pp. 3371–3408, 2010.
- [14] G. E. Hinton, A. Krizhevsky, and S. D. Wang, “Transforming auto-encoders,” in *Artificial Neural Networks and Machine Learning – ICANN 2011*, T. Honkela, W. Duch, M. Girolami, and S. Kaski, Eds., 2011, pp. 44–51.
- [15] B. Hou and R. Yan, “Convolutional auto-encoder based deep feature learning for finger-vein verification,” in *2018 IEEE International Symposium on Medical Measurements and Applications (MeMeA)*. IEEE, 2018.
- [16] B. H. Toby and R. B. Von Dreele, “GSAS-II: the genesis of a modern open-source all purpose crystallography software package,” *Journal of Applied Crystallography*, vol. 46, no. 2, pp. 544–549, 2013.
- [17] *NOMAD*, Oak Ridge National Laboratory. [Online]. Available: <https://neutrons.ornl.gov/nomad>
- [18] D. P. Kingma and J. L. Ba, “ADAM: a method for stochastic optimization,” in *International Conference on Learning Representations*, 2015, pp. 1 – 13.