Towards a Universal Classifier for Crystallographic Space Groups

Nouamane Laanait, Junqi Yin, Albina Borisevich Oak Ridge National Laboratory

State of the art electron microscopes produce focused electron beams with atomic dimensions and allow to capture diffraction patterns arising from the interaction of incident electrons with nanoscale material volumes. Backing out the local atomic structure of said materials requires compute- and time-intensive analyses of these diffraction patterns (known as convergent beam electron diffraction, CBED). Traditional analyses of CBED requires iterative numerical solutions of partial differential equations and comparison with experimental data to refine the starting material configuration. This process is repeated anew for every newly acquired experimental CBED pattern and/or probed material.

In this data, we used newly developed multi-GPU and multi-node electron scattering simulation codes on the Summit supercomputer to generate CBED patterns from over 60,000 materials (solid-state materials), representing nearly every known crystal structure. The overarching goals of this data challenge is to: (1) explore the suitability of machine learning algorithms in the advanced analysis of CBED and (2) produce a machine learning algorithm capable of overcoming intrinsic difficulties posed by scientific datasets.

The data set is split across multiple HDF5 files and an accompanying Jupyter Notebook provides a detailed description on how to navigate the file structure to access the data samples and the associated materials properties. Briefly, a data sample from this data set is given by a 3-d array formed by stacking 3 CBED patterns simulated from the same material at 3 distinct material projections (i.e. crystallographic orientations). Each CBED pattern is a 2-d array (512x512 pixels) with float 32-bit image intensities.

Associated with each data sample in the data set is a host of material attributes or properties which are, in principle, retrievable via analysis of this CBED stack. These consists of the crystal space group the material belongs to, atomic lattice constants and angles, chemical composition, to name but a few. Of note is the crystal space group attributed (or label). All possible spatial arrangements of atoms in any solid (crystal) material obey symmetry conditions described by 230 unique mathematical discrete space groups.

The data challenge questions revolve around developing and implementing a machine learning (ML) algorithm to predict a material's space group, essentially a classification task. The data set is, however, heavily imbalanced (i.e. number of data samples per class). This imbalance is not an artifact, instead it reflects the reality that most known materials have low symmetry and as such are not uniformly distributed across the 230 space group classes.

The challenge questions are as follows:

0. Perform exploratory data analysis on both CBED patterns and materials properties to summarize data characteristics.
1. Develop an ML algorithm for space group classification of CBED data.
2. Implement proper ML techniques to overcome data/label imbalance and show how it affects the performance of the ML algorithm in (1)
3. Implement an ML algorithm for multi-task prediction of a space group in addition to other material structural properties and show how it affects the performance of the ML algorithm in (1).

Notes on the challenge questions:

- Preliminary question (0) is meant to provide better understandings about both input data (e.g. principle components of input images) and targets (e.g. distribution of space groups)
- A participant may choose to do (0), (1) and (2) or (0), (1) and (3). Completing all 4 questions is optional.
- Regarding approaches to (2), our preference is for ML techniques (e.g. loss-weighting, model ensembles, active learning, decision boundary analysis with GANs, etc…), in lieu of brute-force data augmentation approaches (e.g. mixup, random erasing, etc…).
- If a deep learning model is used by the participant our preference is for the implementation to use one of the following 3 frameworks: MXNet, Pytorch or TensorFlow.
- Our preference is for the ML algorithms to be implemented in one of the following languages: Python, C/C++, Julia.