Image Segmentation for Automated Sample Alignment in Neutron Scattering Experiments

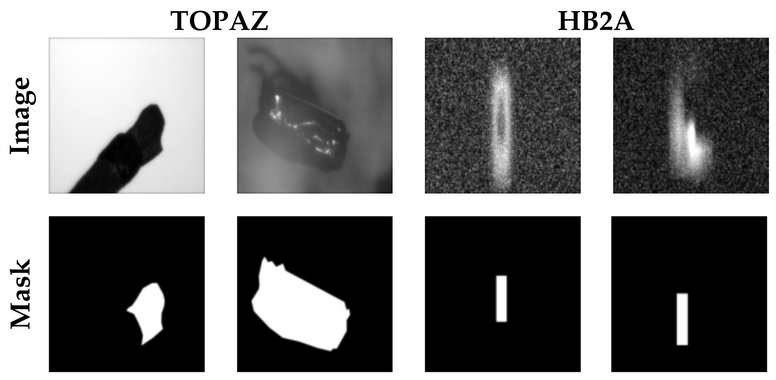
J.P. Edelen*, **M. Henderson**, M. Kilpatrick, I. Pogorelov (RadiaSoft LLC, Boulder, USA);
S. Calder, C. Hoffman, B. Krishna, B. Vacaliuc (Oak Ridge National Laboratory, Oak Ridge, USA);
*jedelen@radiasoft.net

Abstract

The use of machine learning models, particularly convolutional neural networks (CNNs), has become a favored technique for processing image data in many research contexts. One area in which such models can be leveraged to streamline workflows for neutron scattering experiments is the automation of sample alignment protocols. Unlike applications which require only the detection or classification of objects, identifying sample centers of mass used for alignment requires image segmentation which describes both the location and spatial extent of a sample. RadiaSoft is currently working with scientists at Oak Ridge National Laboratory (ORNL) to develop modular, customizable software which interfaces with machine-level beamline data, processes that data, and returns alignment information to beamline controls. In this talk, we provide an overview of our interface software and segmentation models. We also discuss the application of our models to images from different beamlines, the extension of pre-trained models using transfer learning, and methods for quantifying model uncertainty.

Sample Images & Masks

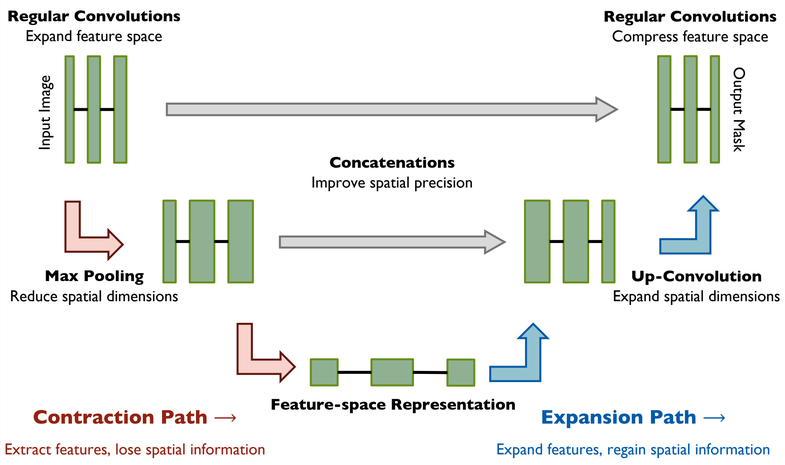
The images of scattering samples used in this project are collected at the TOPAZ [1] and HB2A [2] beamlines at ORNL. Images from TOPAZ are taken by ordinary cameras, but may be taken alongside or from below the sample depending on the operational mode. Images from HB2A are taken using a neutron camera, and feature high levels of noise. Masks used during network training must be defined manually.



Example beamline sample images & their manually defined masks

CNN Architecture

The CNN architecture used for image segmentation in this project is based on UNet [3]. It features a contraction path which passes inputs through convolutions and max-pooling operations to encoded feature information while reducing spatial dimensions. Convolutions are then performed on the final feature-space representations, further condensing the feature information. This information is passed through an expansion path of regular and up-convolutions to localize feature information and concatenation with expansion path feature maps to improve spatial precision. Our models consist of five convolutional layers, with numbers of filters that begin at 16 and double at each successive depth.



A block-diagram depiction of the general UNet architecture

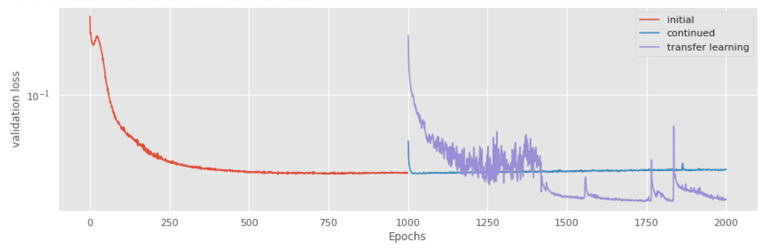
Network Training

Our initial network training efforts involved the construction of UNets for each beamline based on publicly available implementations. The hyperparameters of these models were modified iteratively to yield minimally performant models. Taking these as starting points, we conducted procedural hyperparameter optimization resulting in optimized models with substantially reduced validation losses.

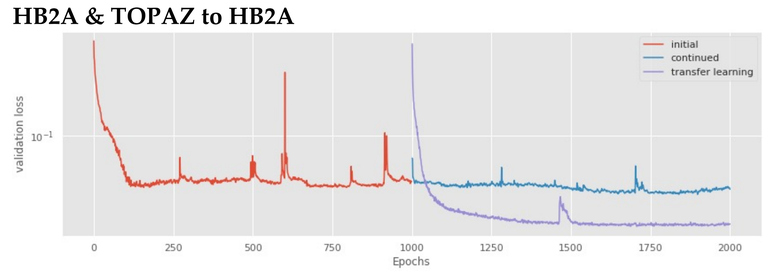
Transfer Learning

A necessary feature of our controls interface software is the ability to rapidly deploy to new beamlines or sites. To test the extensibility of our models, we performed additional rounds of training during which optimized models were re-trained using either the same data used during initial training or underwent transfer learning [4, 5] on data from the opposite beamline. This provides an especially robust test of model extensibility given the alternation between ordinary and neutron camera images. We found that transferred models not only achieve similar validation losses compared to those trained on a single dataset, but ultimately outperformed them regardless of the order of training.

TOPAZ & HB2A to TOPAZ



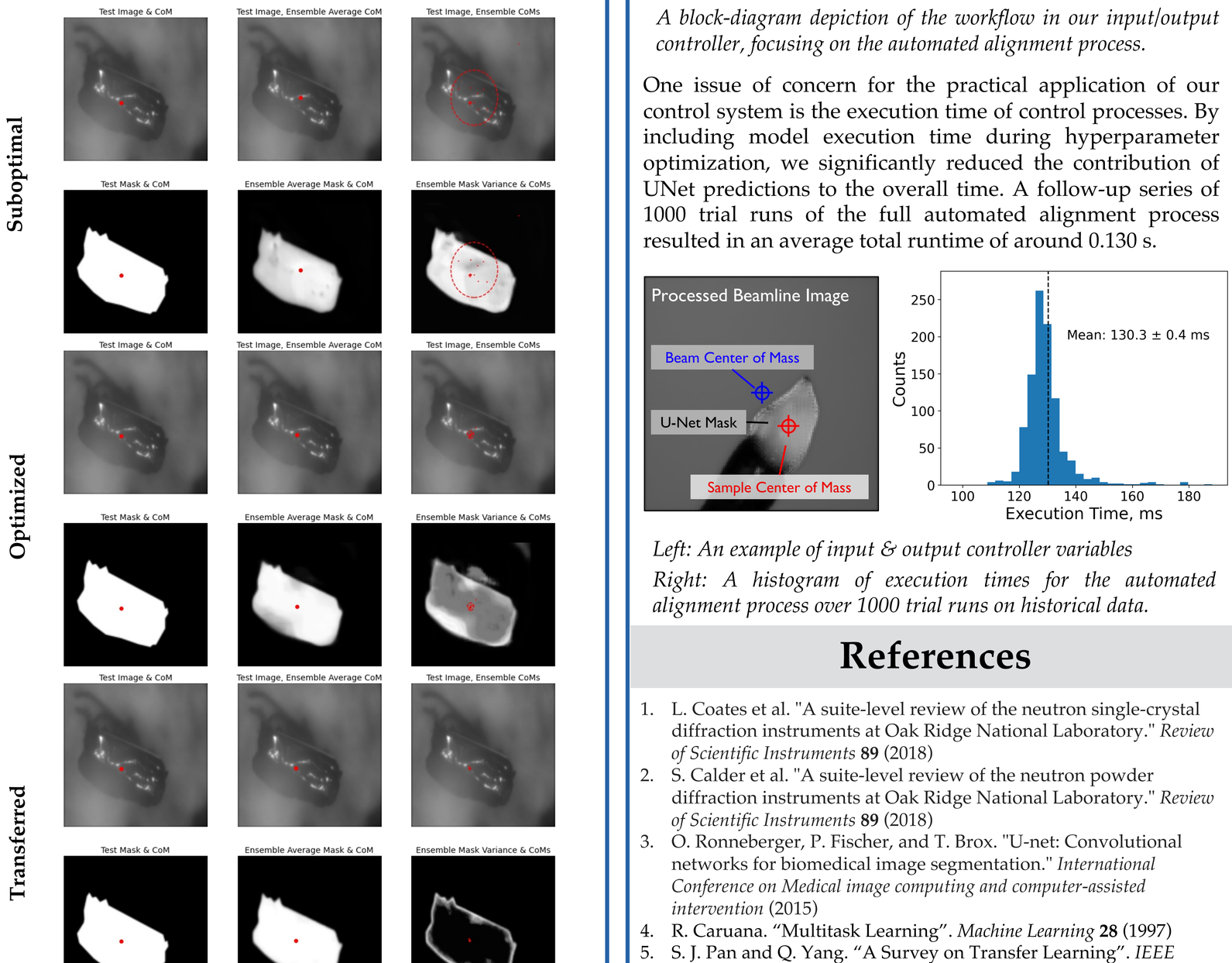
HB2A & TOPAZ to HB2A



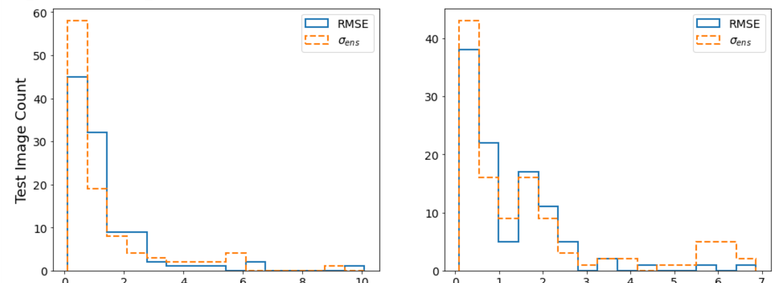
Validation losses for continued (blue) and transferred (purple) training of models initially trained (orange) one set of data

Uncertainty Quantification

Accurate, quantitative uncertainty estimation is paramount for any ML application which will eventually inform real-world decisions. In our control system, uncertainties will be used to determine, e.g., the need for human intervention during alignment procedures. To quantify model prediction uncertainty, we randomly initialized and trained ensembles of 20 models for 16 distinct combinations of training set (TOPAZ/HB2A), hyperparameter optimization (or lack of), and continued or transferred training (if any). Computing variances in the mask and center of mass predictions of each model in an ensemble over a validation set, we were able to compare against the MSE metric which is used during training and confirm that the distributions of both metrics were (in all cases) closely comparable. This is a highly useful result as the MSE cannot be computed during live operation and aleatoric measures of uncertainty like the ensemble variance do not always provide a good approximators of epistemic quantities like the MSE [6].



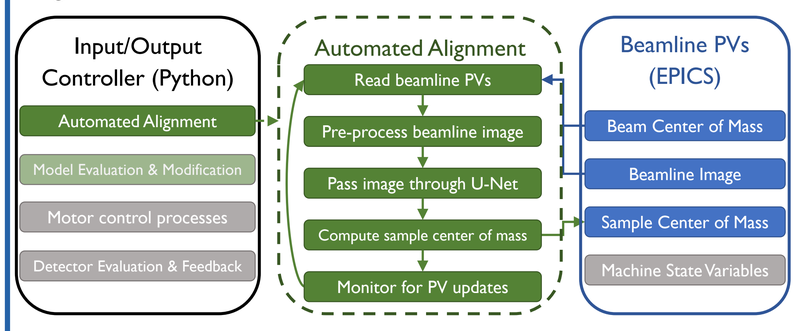
Results uncertainty quantification through ensemble statistics, including predictions for the sample mask and center of mass.



Distributions of RMS error and ensemble variances for an ensemble of optimized models trained on TOPAZ images.

Control System

The control system into which our machine learning models are embedded essentially functions as an interface between machine-level controls managed using the EPICS package and user-defined alignment processes. Quantities of interest such as image data, beam center positions, and sample centers of mass are monitored, processed, and modified by a central Python program and exist locally as EPICS process variables (PVs). This allows us to read sample image data, preprocess them, and pass them through our image segmentation models in real time.



A block-diagram depiction of the workflow in our input/output controller, focusing on the automated alignment process.

One issue of concern for the practical application of our control system is the execution time of control processes. By including model execution time during hyperparameter optimization, we significantly reduced the contribution of UNet predictions to the overall time. A follow-up series of 1000 trial runs of the full automated alignment process resulted in an average total runtime of around 0.130 s.

Left: An example of input & output controller variables
Right: A histogram of execution times for the automated alignment process over 1000 trial runs on historical data.

References

1. L. Coates et al. "A suite-level review of the neutron single-crystal diffraction instruments at Oak Ridge National Laboratory." *Review of Scientific Instruments* **89** (2018)
2. S. Calder et al. "A suite-level review of the neutron powder diffraction instruments at Oak Ridge National Laboratory." *Review of Scientific Instruments* **89** (2018)
3. O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation." *International Conference on Medical image computing and computer-assisted intervention* (2015)
4. R. Caruana. "Multitask Learning". *Machine Learning* **28** (1997)
5. S. J. Pan and Q. Yang. "A Survey on Transfer Learning". *IEEE Transactions on Knowledge and Data Engineering* **22** (2010)
6. E. Hüllermeier and W. Waegeman. "Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods". *Machine Learning* **110** (2021)

U.S. DEPARTMENT OF ENERGY | Office of Science

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences, SBIR and STTR Program under Award Number(s) DE-SC0021555. The standard DOE disclaimer applies to any content funded by DOE: github.com/radiasoft/public/wiki/DOE-Disclaimer

ICFA Mini-Workshop on Machine Learning Applications for Particle Accelerators
Chicago, IL. 11/01/2022 – 11/04/2022

radiasoft
Boulder, CO USA | radiasoft.net