Predicting the Geometry of X-Ray Crystallography Diffraction Images Using Computer Vision

David Lin[1], Azfal Peermohammed[1], Jade Thai[1], Haoran Yan[1], Xinshu Zhao[1], Sichen Zhong[1], Laura Baracaldo[1], Derek Mendez [2]

[1] University of California, Santa Barbara; [2] Stanford National Accelerator Lab

SLAC NATIONAL ACCELERATOR LABORATORY

UCSB Data Science Capstone

UC SANTA BARBARA | Data Science Initiative

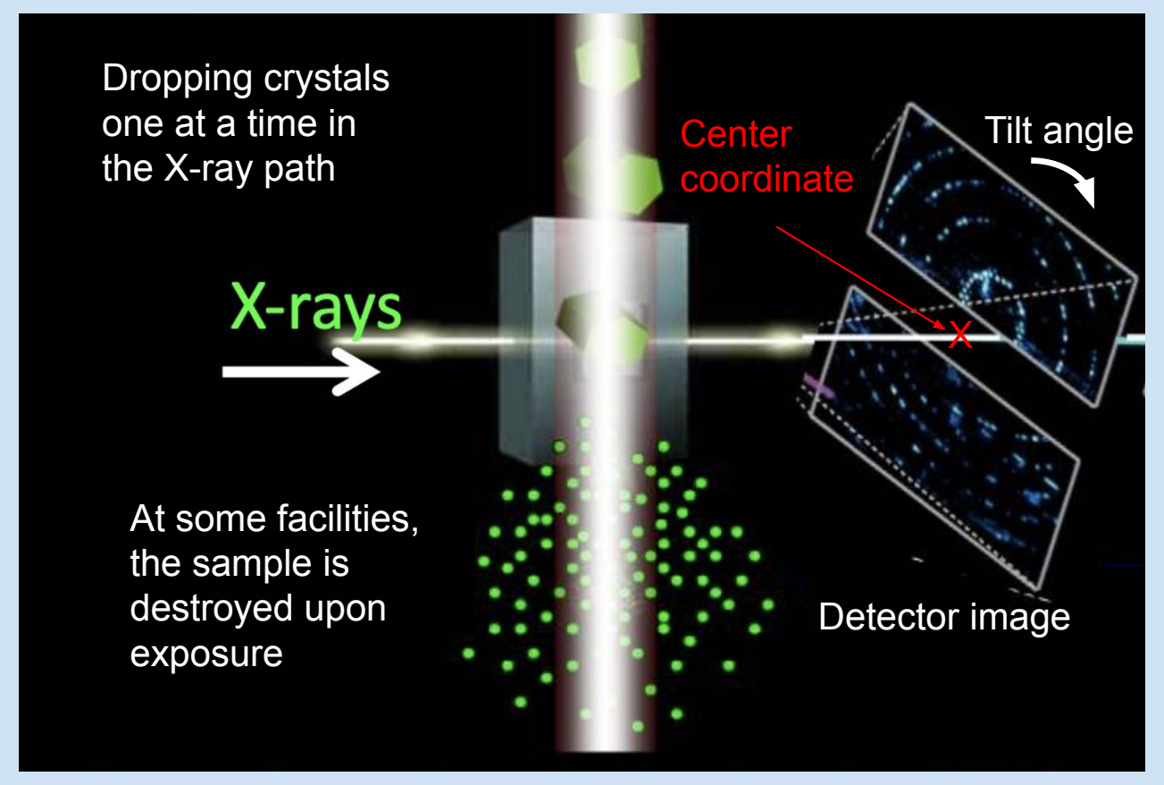
Abstract

In protein X-ray crystallography, protein crystals are exposed to X-ray beams, and the resulting diffraction patterns are recorded on two-dimensional images. By processing these diffraction images, researchers can obtain atomic-scale information about the protein's structure. This is invaluable for our understanding of how proteins operate in fields like therapeutics and energy production. Before we can extract any information about the structure of the protein, we must first understand the geometry of the diffraction images utilizing structures called Bragg peaks. Two quantities of interest are the so-called "beam center" and the "detector tilt". The beam center is the point where the incident X-ray beam vector intersects the camera, and the tilt is the angle the detector face makes relative to a plane perpendicular to the incoming X-rays. The goal of this project will be to use artificial intelligence to process X-ray diffraction images to determine beam center and tilt.

Introduction

X-ray crystallography is a scientific technique used to determine the atomic structure of crystalline materials; in this case, the crystalline structures are proteins. By exposing the crystal to X-ray radiation, scientists can measure the scattering angles and intensities of the diffracted beams, ultimately revealing the arrangement of atoms within the crystal lattice. This information is critical for understanding the properties and behaviors of materials at the atomic level, which is beneficial for many scientific fields, including chemistry, biology, materials science, and physics. The crystal diffraction images contain regions of interest called Bragg peaks which can be seen in *Fig 1* as the bright dots on the image. Bragg peaks can also be seen as black dots in the images depicted in *Figs 2-6*.

Fig 1: Depiction of X-Ray crystallography and the creation of diffraction images. The center is shown as the red X (the point where incident X-ray vector intersects the detector image). The tilt is parameterized by the angle(s) the detector makes in relation to the incoming X-rays.


Dropping crystals one at a time in the X-ray path
Center coordinate
Tilt angle
X-rays
At some facilities, the sample is destroyed upon exposure
Detector image

Centering: An essential feature of the diffraction geometry is the precise sub-pixel coordinate of the point where the incident X-ray beam intersects with the detector plane. This is shown visually by the red X in Figure 1. We believe that the Bragg peaks alone will provide sufficient information to predict this coordinate in individual images.

Detector Tilt: In the context of X-ray protein crystallography, the orientation of the detector can be described by a three-dimensional rotation matrix (three angles). For simplicity, we will be training a model to predict two angles of rotation about orthogonal axes, the pitch and yaw, which together correspond to the overall detector tilt. If we assume the incident X-rays travel along the +z direction, then the pitch, yaw are rotations of the detector about the +x, +y directions, respectively. The pitch angle can be described as "nose up" and "nose down" and the yaw angle as "left" and "right" in airplane motion.

Datasets

- All of the data follow the HDF5 (Hierarchical Data Format version 5) format which is a versatile file format and computational library designed for managing large and complex datasets.
- The training data consists of roughly 110,000 downsampled (1x820x820) simulated 2D diffraction images, while the actual images are 1x2527x2463.
- The labels are 2-element vectors representing the X-ray beam center coordinate (pixels) or the tilt of the detector (degrees).
- The datasets vary in beam center location, detector tilt, detector distance, background noise, crystal morphology, and image formatting including grayscale and binary images.

Representative diffraction images in greyscale (black is higher intensity). Note the Bragg peaks are shown as black dots. The centrosymmetric features (diffuse, ring-like features) represent scattering from amorphous materials like water, present in most experiments. We call these background X-ray scattering.

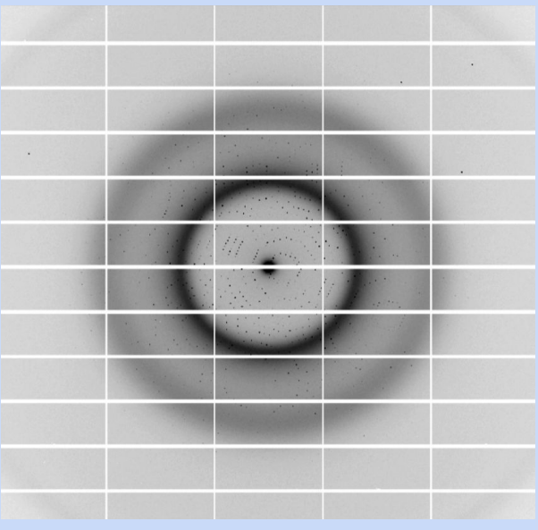


Fig 2: Standard diffraction with average sample-to-detector distance.

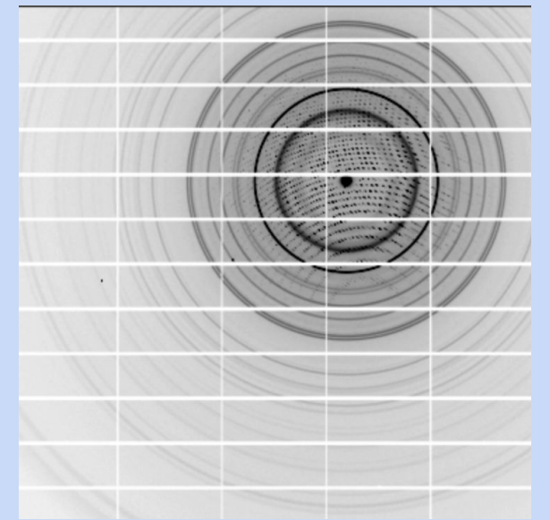


Fig 3: Diffraction with widely varied center

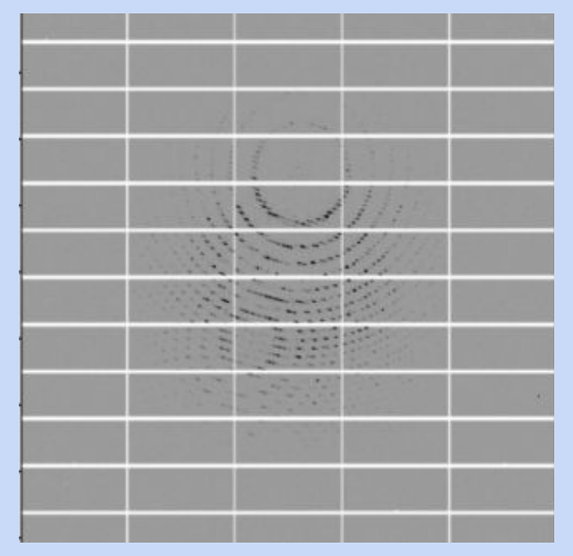


Fig 4: Diffraction image with flat background

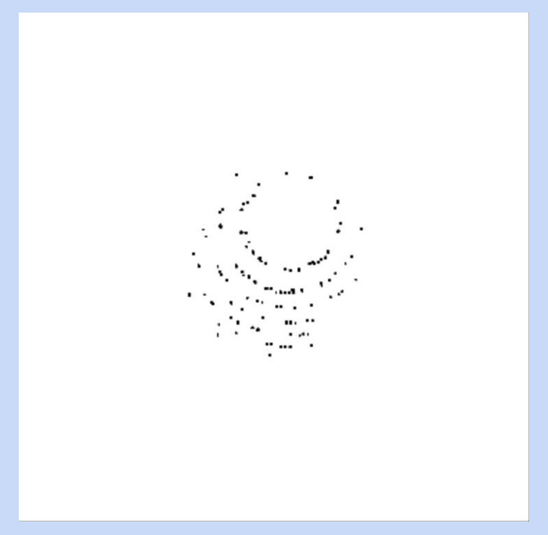


Fig 5: Binary diffraction image (Bragg peaks have value 1)

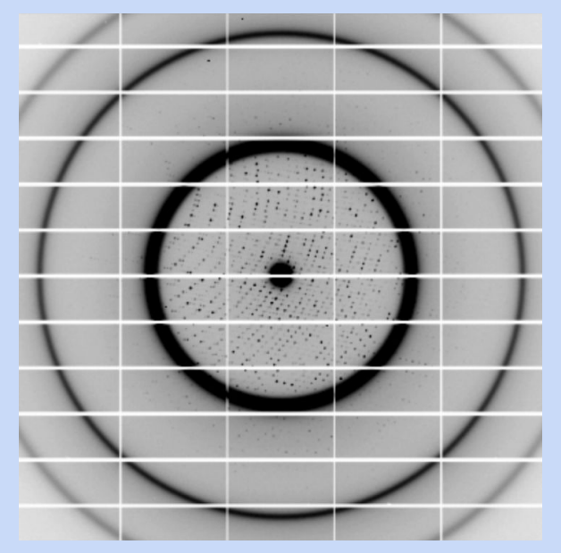


Fig 6: Diffraction with longer sample-to-detector distance (background is more expanded)

Methodology

- The primary computational library we will be using for our project is PyTorch.
- We leveraged the computational power of HPC through the UCSB Center for Scientific Computing, training on NVIDIA 64-bit V100 GPUs.
- For experiment tracking we utilized TensorBoard, a feature from the package TensorFlow to track the loss of all models.
- Both groups began by training data sets on a baseline model, which consisted of a simple CNN. We then compared the results to various ResNet architectures, as described below:

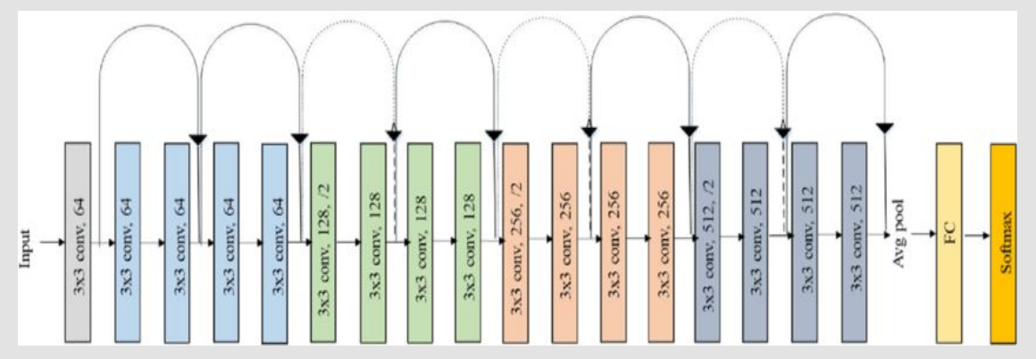
Baseline Model

- Consists of four convolutional layers (conv)
- Two max-pooling layers which downsample the image by a factor of two after the first conv layers
- A fully connected layer (FC) preceding the outputs

ResNet 18

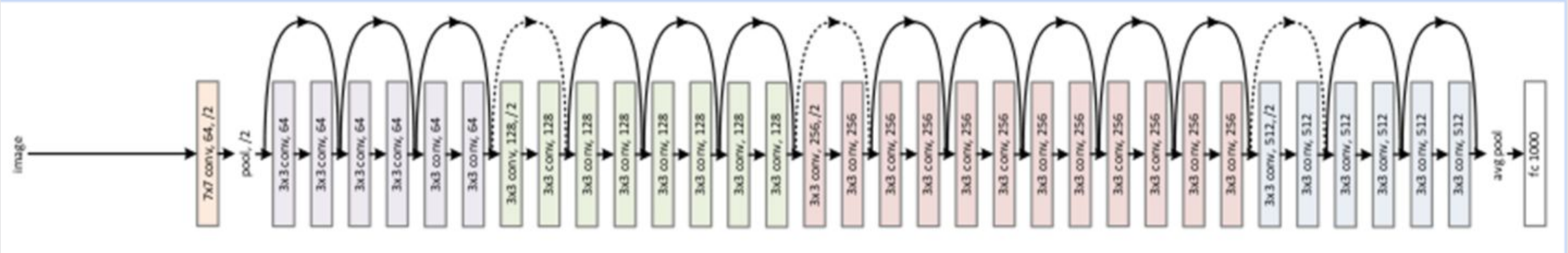
- Consists of eight residual blocks
- Each block contains two conv layers where the activations prior to the residual block become concatenated to the activations from the second layer before fed into the next block

Fig 7: Model architecture of Resnet 18



Larger Resnet models simply consists of more residual blocks.

Fig 8: Model architecture of Resnet 34 model



Results

Centering:

- The ResNet architectures outperformed the baseline model's loss of about 3 pixels.
- We found that after tuning hundreds of variants of the ResNet, a learning rate of 0.05, 300 activations, and using stochastic gradient descent led to the best models with a validation loss of 0.288.
- We tested our trained model on images with flat backgrounds (*see Fig 6*) and found poor performance. This discovery informed us that the models were using the background scattering to predict the center and not the bragg peaks as intended.

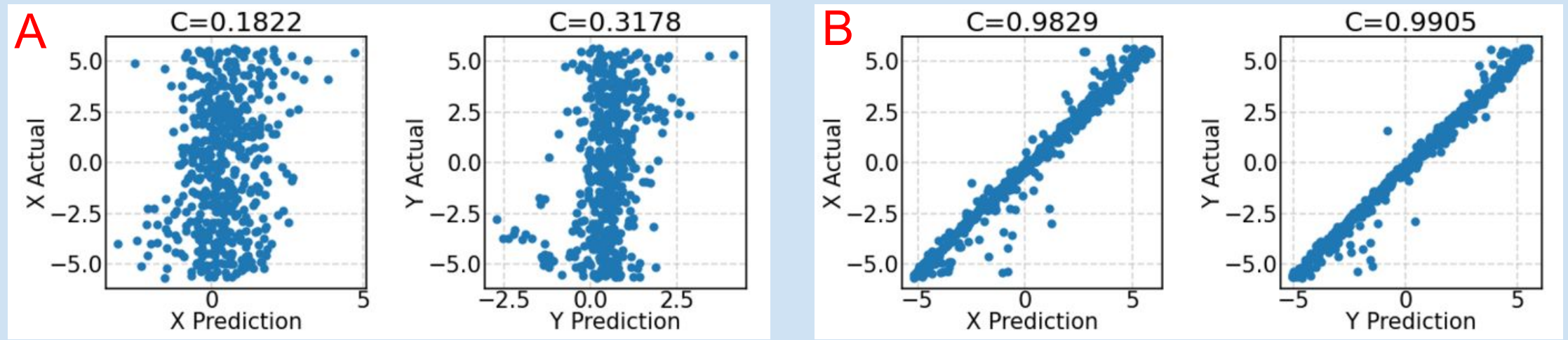


Fig 9: After training a model on the standard images (e.g., Fig 2), we show the actual versus predicted coordinates for (A) images with flat backgrounds (like Figure 4), and (B) images akin to those used during training (e.g., Figure 2). All units are in Pixels, and Pearson correlations are shown for reference. These results indicate the model learns the center by analyzing the background scattering.

- Based on the idea that background scattering could be sufficient to predict the center, we trained a model using training data with increased variability in the backgrounds (*e.g., Fig 6*).

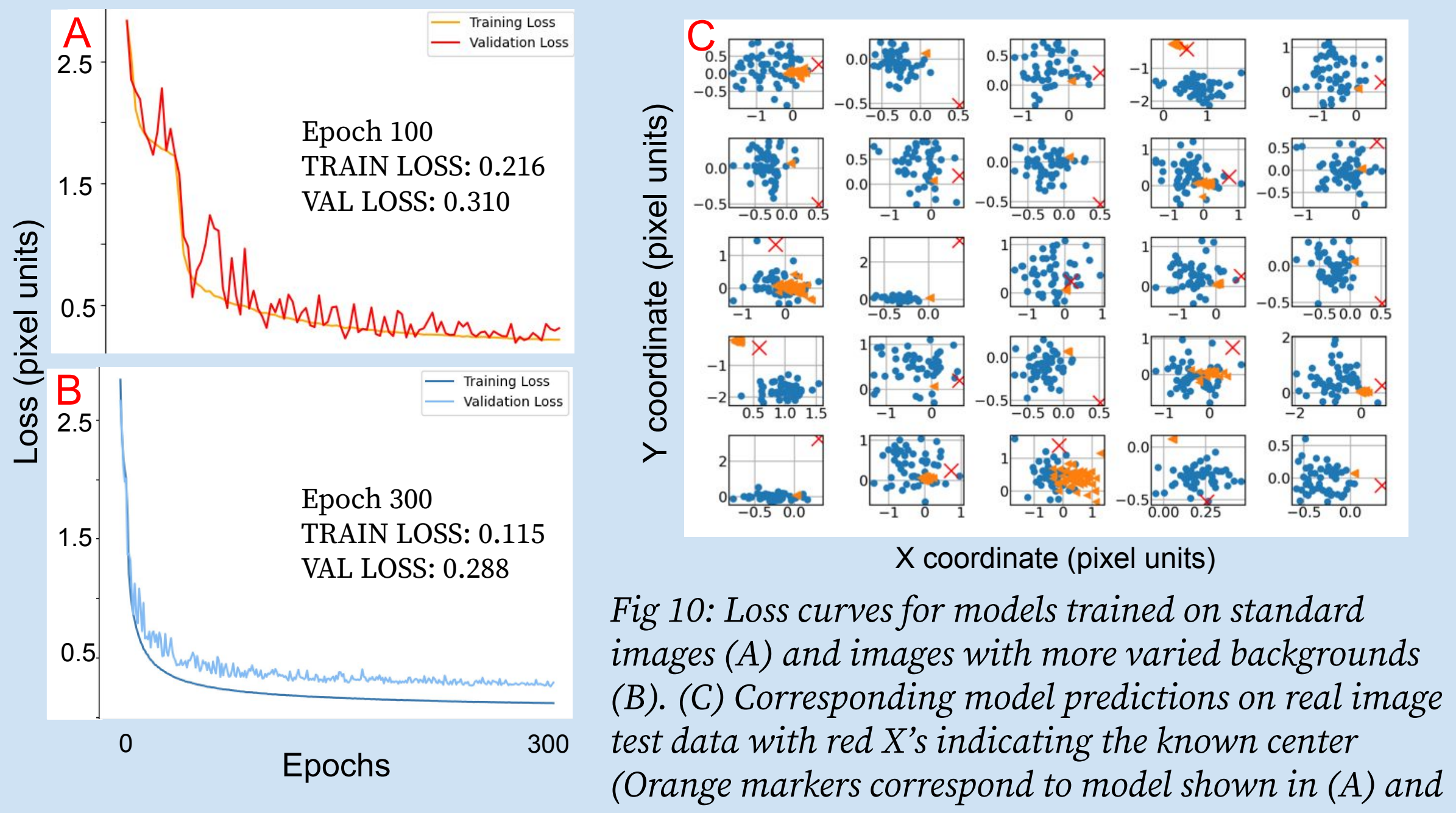


Fig 10: Loss curves for models trained on standard images (A) and images with more varied backgrounds (B). (C) Corresponding model predictions on real image test data with red X's indicating the known center (Orange markers correspond to model shown in (A) and blue markers correspond to model shown in (B)).

Tilting:

- We trained a variety of ResNet models. The images in the training dataset varied with tilt angles ranging from -3 degrees to 3 degrees.

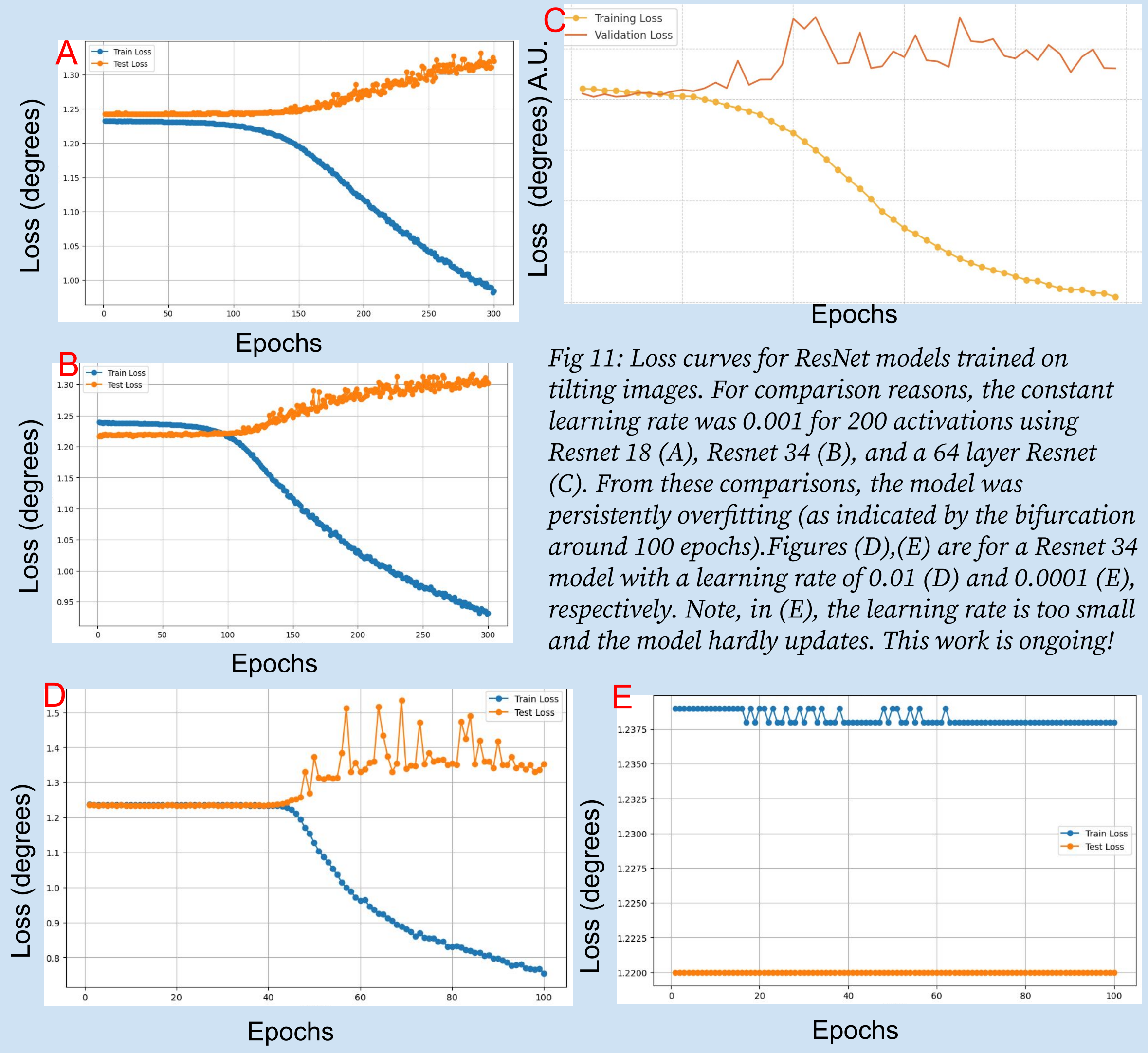


Fig 11: Loss curves for ResNet models trained on tilting images. For comparison reasons, the constant learning rate was 0.001 for 200 activations using Resnet 18 (A), Resnet 34 (B), and a 64 layer Resnet (C). From these comparisons, the model was persistently overfitting (as indicated by the bifurcation around 100 epochs).Figures (D),(E) are for a Resnet 34 model with a learning rate of 0.01 (D) and 0.0001 (E), respectively. Note, in (E), the learning rate is too small and the model hardly updates. This work is ongoing!

Summary of findings

Centering:

- We found that out of all the ResNet models, ResNet 18 seemed to be the best on predicting the center coordinates.
- When testing on real images, the model struggled to correctly identify the center as determined by conventional processing tools.
- We discovered that the ResNet was picking up on background scattering, contrary to our initial goal of using the Bragg peaks.
- The predictions on the testing data tended to be smaller than the actual labels along the x-axis.
- Inaccuracy is attributed to the type of the simulated data that the models were trained on.

Tilting:

- The loss values of ResNet-34 is slightly less than ResNet-18, which is about 1.295. Ideally, the error would be much smaller (~0.1 degrees).
- Specifically, we observed strong over-fitting to the training data, so further tuning of model is required to achieve higher accuracy.
- The performance of the degree prediction on yaw-axis is better than on pitch-axis on every model since they have different distribution.
- We believe ResNet-50 with appropriate learning rate (maybe 1e-3) holds promise for increasing accuracy.

Future work

- If allotted more time we would continue testing models on even more data until we obtain one which can pick up on the two center coordinates in the real da ta.
- Incorporate further parallelization with training on multiple GPUs to decrease the training runtime, allowing more experimentation with parameters.
- Create a pipeline to determine what images our models struggle with the most.
- Explore even more architectures, like Transformer models.

References and Acknowledgments

[1] *Deep Residual Networks for Crystallography trained on Simulated Data*, Acta Crystallogr D Struct Biol. (2024)

[2] *Deep Residual Learning For Image Recognition*, arXiv:1512.03385 (2015)

We would like to thank SLAC National Accelerator Laboratory for sponsoring this project as well as the UCSB Center for Scientific Computing. A special thanks to Derek Mendez, Laura Baracaldo, and Jay Chi for all their guidance and support.

Github Repository