# Extremely Noisy 4D-TEM Strain Mapping Using Cycle Consistent Spatial Transforming Autoencoders

**Shuyu Qin**
Department of Computer Science
Lehigh University
Bethlehem, PA 18018
shq219@lehigh.edu

**Nhan Tran**
Fermilab
Batavia, IL 60510
ntran@fnal.gov

**Joshua Agar**
Department of Mechanical Engineering and Mechanics
Drexel University
Philadelphia, PA 19104
jca92@drexel.edu

## Abstract

Atomic-resolution imaging of 2D and quantum materials benefits from precisely extracting crystallographic strain, shear, and rotation to understand their mechanical, optical, and electronic properties. One powerful technique is 4-D STEM (4-dimensional scanning transmission electron microscopy) [1], where a convergent electron beam is scanned across a sample while measuring the resulting diffraction pattern with a direct electron detector. Extracting the crystallographic strain, shear, and rotation from this data relies either on cross-correlation of probe templates (e.g., implemented in py4DSTEM [2, 7, 8, 9]) or determining the center of mass (CoM) of the diffraction peaks [10, 11]. These algorithms have limitations. They require manual preprocessing and hyperparameter tuning, are sensitive to signal-to-noise, and generally are difficult to automate. We build novel and unique neural network structure cycle-consistent-spatial-transforming autoencoders (CC-ST-AE) for extracting spatial parameters on noisy 4-D STEM and generate more accurate and robust results compared with state-of-art.

## 1 Introduction

In the 4-D STEM field, whether correctly measuring the crystal structure with sub-atomic precision can strongly affect revealing their mechanical, optical, and electronic properties. It is challenging for current strategies directly working on extremely noisy 4D-STEM strain mapping, the precision cannot be guaranteed.

Recently, machine learning techniques have been used to assist in analyzing 4D-STEM data, however, these models do not possess the capacity to learn the strain, rotation, or translation instead they just learn an approximation that almost always tends to be correct as long as the test examples are within the training dataset distribution.

We developed a novel neural network structure – Cycle Consistent Spatial Transforming Autoencoder (CC-ST-AE, Fig 1). This model takes a set of diffraction images and trains a sparse autoencoder to classify an observed diffraction pattern to a dictionary of learned set of "averaged" diffraction patterns. Secondly, it learns the affine transformation matrix parameters that minimize the reconstruction error between the dictionary and the input diffraction pattern [12]. Since the affine transformation includes

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

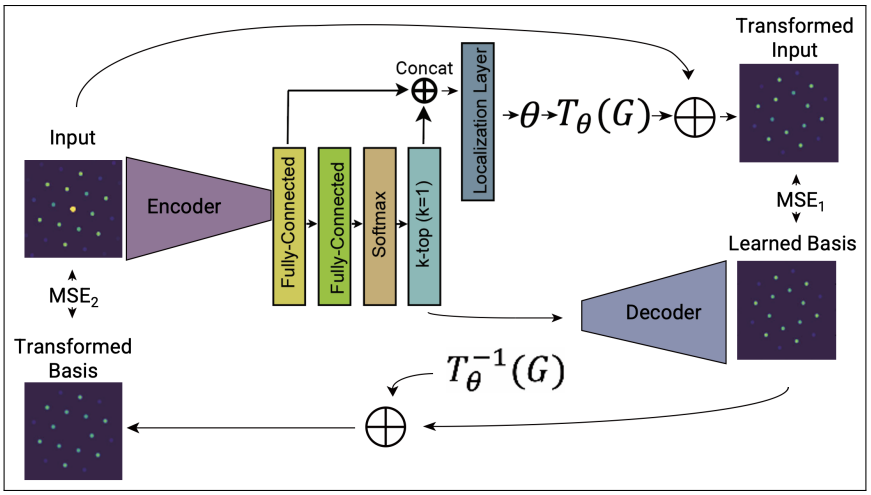


Figure 1: Schematic drawing of cycle-consistent spatial transforming autoencoder.

translation, strain, shear, and rotation, we can parsimoniously learn the strain tensor. To ensure the model is physics-conforming, train the model cycle consistently, by ensuring the inverse affine transformation from the dictionary results in the original diffraction pattern.

We update the weights of the model based on the loss of mean squared error (MSE) between input datasets and output of the decoder. The inputs to the decoder are binary vectors which only have one element activated (equal to 1) and the others deactivated (equal to 0). We need the encoder not only to extract the spatial transform matrix but also to do classification on each diffraction image. The encoder consists of multi-convolutional layers [3], fully connected layers, and one K-Sparse layer [4]. K-Sparse layer only keeps the highest K activities from input vectors - in this case 1. This achieves exact sparsity in the hidden representation which is used for classification in our model structure. The diffraction images belonging to the same cluster generate the same binary vector through the encoder which results in the same base through the decoder. Comparing the base with input images belonging to it can make the training process focus solely on learning the spatial transformation. To enhance this training strategy, we train the model cycle consistently.

Besides MSE in loss function, we also add regularizations to make the model train efficiently and correctly. We add L1 regularization on learned bases to prevent the learning process from focusing on extremely high-intensity values and make the generated bases sparse. The regularization of strain and shear parameters is also added to make the learned affine parameters change in a reasonable range. What we care about is the position of diffraction spots, to prevent extending beyond the boundary and avoid the high-intensity center spot from being considered by the model training process, we apply a mask region to each diffraction image. We make a circular ring mask to prevent the training from being influenced by irregular boundary shapes caused by various rotation parameters. Last but not least, when training on extremely noisy 4D-STEM, it is hard to get robust results since the intensity of background noise and diffraction spots are too similar to distinguish. Therefore we slowly reduce the size of the mask, minimizing the influence of the background noise and eliminating learning the strain of the noise profile. Thus we minimize the effects of noise focusing on the important information contained in the diffraction spot. However, such strategy has the limitation that it is impossible to block all the noise, and it is also hard to guarantee whole diffraction spots are included when reducing the size of mask. Different affine parameters applied on images will vary the size of noise, which makes model optimize noise size and results in poor performance. To minimize these effects, inverse affine transformation are added for every pixel in mask region.

We validated this model on a number of benchmark tasks including Simulated 4D-STEM data with a ground truth of the strain, rotation, and shear parameters. Then, applied different levels of poisson background noise on simulated 4D-STEM to test the robustness of the model and compare it with other strategies. Secondly, we test this model experimental 4D-STEM on two different tungsten disulfide (WS2) and tungsten diselenide (WSe2) 2D-heterostructures [5].

The open-source computational tool – py4DSTEM provides high-throughput multimodal data correlation strain mapping tool to the community. Which is a powerful tool and broadly used for extracting atomic-scale information from 4D-STEM datasets. Therefore, we list py4DSTEM as the baseline to compare with our method during the training process. To evaluate the performance and robustness of each strategy, we first train the model on the simulated 4D-STEM [6]. The simulated 4D-STEM consists of different sizes of grains connected with each other, the strain, shear, and rotation parameters remain the same in each grain and vary among them. The strain and shear range from -2% to 2% and rotation from 0 to 360 degrees. The goal is to extract these parameters as accurately as possible.

Recently, we developed an open-source computational tool – Auto4DSTEM, which to analyze 4D-STEM datasets. It includes various types of custom CC-ST-AE neural networks for extracting different atomic-scale information depending on the dataset. The tool is multi-functional, including data preprocessing, model architecture design, and neural network training with hyperparameter tuning, which covers every step from raw data to final results. The notebook of tutorial on simulated 4D-STEM is included in the supplemental material.

## 2 Results and Discussion

We start from noise-free 4D-STEM and use MAE (Mean Absolute Error) between the label and learned spatial parameters for performance evaluation. Both strategies can generate results of MAE below $2 \times 10^3$ (10%) difference compared with the label, our model can make an average error around $3 \times 10^{-4}$ lower, however this difference is largely inconsequential. The noise-free 4D-STEM is an unrealistically easy test as all experimental datasets have noise. To simulate the experimental 4D-STEM, the various intensity of poisson-distributed background noise is added to the dataset to mimic the real background noise created by an electron beam. The percentage of poisson noise intensity is set from 5% to 70%, we train the dataset at each level of noise intensity and compare the training results between py4DSTEM and CC-ST-AE. When adding 5% of poisson distributed background noise, there's a clear change for py4DSTEM, the MAE increased by 20% compared with the noise-free dataset. However, the CC-ST-AE strategy can maintain similar performance. When increasing the intensity of background noise, the MAE and growth rate of MAE are both higher for py4DSTEM compared with CC-ST-AE. When the intensity of background noise goes to 50%, the MAE of Strain X (strain along horizontal direction) is $3.2 \times 10^{-3}$ for py4DSTEM, which gets 60% worse than the performance on the noise-free dataset. The MAE is $1.8 \times 10^{-3}$ for CC-ST-AE, which performs more than 43% better than py4DSTEM, only 20% worse than itself on the noise-free dataset. The MAE of Strain Y (strain along vertical direction) is $3.2 \times 10^{-3}$ for py4DSTEM, which is 60% worse than the performance on the noise-free dataset. The MAE is $1.9 \times 10^{-3}$ for CC-ST-AE, which performs more than 40% better than py4DSTEM, only 5% worse than itself on the noise-free dataset. The MAE of Shear is $1.9 \times 10^{-3}$ for py4DSTEM, whereas the MAE is $1.5 \times 10^{-3}$ for CC-ST-AE. Both on strain and shear, CC-ST-AE can perform far better than py4DSTEM. If we only look into strain parameters, the performance of CC-ST-AE on 50% background noise 4D-STEM is even better than the performance of py4DSTEM on noise-free 4D-STEM.

We take 25% background noise 4D-STEM as an example, the map and histogram of strain x (along horizontal direction), strain y (along vertical direction), and strain xy (shear, along diagonal direction) extracted by py4DSTEM and CC-ST-AE are shown in Fig 2. it is clear that CC-ST-AE can extract sharper and more clustered distributions than py4DSTEM on every type of strain. This means less MAE difference compared with the label and more accuracy in physics.

The previous training has revealed that the CC-ST-AE is competitive in spatial transformation matrices extraction and maintains robustness when background noise varies. The table of MAE between results learned by different methods (py4DSTEM and CC-ST-AE) and labels on various levels of Poisson Distributed simulated 4-D STEM showed in Fig 3. We can see that from noise-free 4-D STEM to very high-intensity background noise (70%) 4-D STEM, CC-ST-AE generates better and more robust results compared with py4DSTEM, which is considered to be state-of-the-art. When adding noise intensity until background noise intensity exceeds diffraction spots, the py4DSTEM starts to fail. CC-ST-AE still works at that point (around 60% background noise). The MAE of Strain X for py4DSTEM on 60% background noise gets to $3.9 \times 10^{-3}$, where the MAE for CC-ST-AE is $2.0 \times 10^{-3}$, which performs more than 95% better than py4DSTEM. The MAE of Strain Y for py4DSTEM gets to $4.0 \times 10^{-3}$, where the MAE for CC-ST-AE is $2.1 \times 10^{-3}$, which performs more than 90% better than py4DSTEM. The MAE of Shear for py4DSTEM gets to $2.5 \times 10^{-3}$ whereas

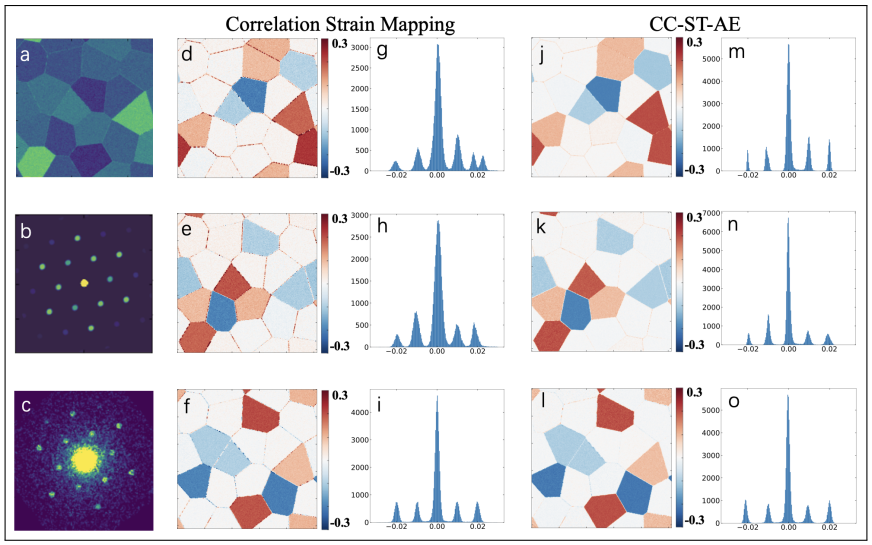


Figure 2: Results comparison between py4DSTEM (correlation strain mapping strategy) and CC-ST-AE on 25% noisy simulated 4D-STEM. a) Average intensity of real-space domain, every pixel corresponds to a diffraction image (b, c), diffraction images in the same grain share the same spatial transformation parameters. b) A sample of diffraction image from noise-free 4D-STEM. c) Add 25% poisson distributed background to b. d, e, f) Map of strain x (horizontal), strain y (vertical), and strain xy (diagonal) of 25% noisy 4D-STEM generated by py4DSTEM. g, h, i) Histogram of strain x (horizontal), strain y (vertical), and strain xy (diagonal) of 25% noisy 4D-STEM generated by py4DSTEM. j, k, l) Map of strain x (horizontal), strain y (vertical), and strain xy (diagonal) of 25% noisy 4D-STEM generated by CC-ST-AE. m, n, o) Histogram of strain x (horizontal), strain y (vertical), and strain xy (diagonal) of 25% noisy 4D-STEM generated by CC-ST-AE.

| BKG noise intensity /Percentage | Strain X (MAE with Label) *1e-3 | | Strain Y (MAE with Label) *1e-3 | | Shear (MAE with Label) *1e-3 | |
|---|---|---|---|---|---|---|
| | py4dstem | CC-ST-AE | py4dstem | CC-ST-AE | py4dstem | CC-ST-AE |
| 0 | 2.0 | **1.5** | 2.0 | **1.8** | 0.9 | **0.7** |
| 5% | 2.3 | **1.5** | 2.4 | **1.6** | 1.2 | **1.2** |
| 10% | 2.3 | **1.5** | 2.4 | **1.6** | 1.3 | **1.0** |
| 15% | 2.3 | **1.6** | 2.4 | **1.7** | 1.3 | **0.9** |
| 20% | 2.4 | **1.7** | 2.5 | **1.7** | 1.4 | **1.0** |
| 25% | 2.5 | **1.6** | 2.5 | **1.9** | 1.4 | **1.1** |
| 30% | 2.5 | **1.6** | 2.6 | **1.7** | 1.4 | **1.2** |
| 35% | 2.6 | **1.8** | 2.7 | **1.8** | 1.5 | **1.1** |
| 40% | 2.7 | **1.7** | 2.7 | **1.8** | 1.6 | **1.4** |
| 45% | 2.8 | **1.8** | 2.9 | **2.0** | 1.7 | **1.2** |
| 50% | 3.2 | **1.8** | 3.2 | **1.9** | 1.9 | **1.5** |
| 60% | 3.9 | **2.1** | 4.0 | **2.4** | 2.5 | **1.6** |
| 70% | 5.4 | **2.2** | 5.5 | **2.6** | 3.6 | **2.1** |

Figure 3: Results comparison of MAE of Strain X, Strain Y and Shear between different methods (py4DSTEM and CC-ST-AE) and label on different levels of poisson distributed background noise simulated 4-D STEM.

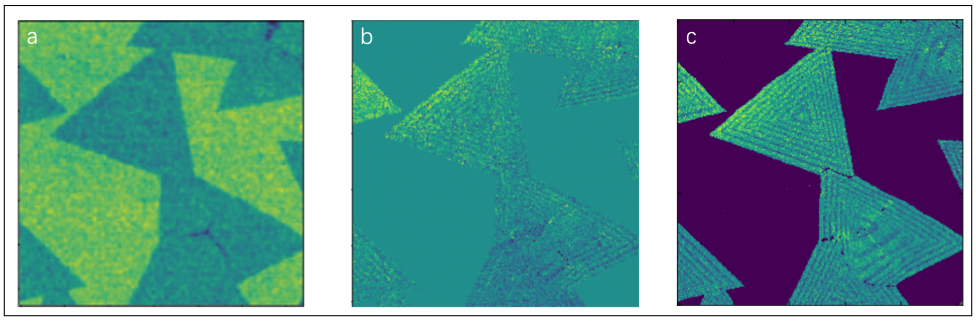


Figure 4: Results comparison between py4DSTEM and CC- ST-AE on $WS_2WSe_2$ 4-D STEM. a) Microscope Image of Sample Domain. b) Strain map image of py4DSTEM result. c) Strain map image of CC-ST-AE result.

the MAE for CC-ST-AE is $1.5 \times 10^{-3}$, which performs more than 75% better than py4DSTEM. Py4DSTEM cannot generate a clear and reasonable distribution on the histogram of the strain map when background noise goes to 60% where CC-ST-AE can. After testing, CC-ST-AE can even work on 70% intensity of background noise 4D-STEM and generate a robust result, which can be considered as an extremely noisy dataset. Furthermore, after the comparison, if we only look into strain parameters, the performance of CC-ST-AE on 60% background noise 4D-STEM has the similar performance of py4DSTEM on noise-free 4D-STEM.

After proving CC-ST-AE can extract more accurate physics and achieve better performance compared with py4DSTEM, we validate the model on experimental 4D-STEM two different tungsten disulfide (WS2) and tungsten diselenide (WSe2) 2D-heterostructures. According to the theoretical character of the material, there should be many triangle shapes on the strain map image. Traditional strategies cannot directly work on extremely noisy raw 4D-STEM with low-intensity diffraction patterns. A common strategy is binning the diffraction image to improve the signal-to-noise but reduces spatial resolution. The result of the strain map created by the neural network is shown in Fig 4, c. To make the comparison, the result of py4DSTEM is shown in Fig 4, b. Since the model can distinguish the sample region from the background, we only focus on the sample and make the background region blank. Both techs can extract strain parameters and generate triangle shapes. Comparing the images, we can find that there are fewer broken regions on the image created by the neural network. it is easy to conclude that the image quality of the neural network is higher than that on py4DSTEM. The training results proved that the CC-ST-AE strategy can work on experimental 4d-stem and have the ability to beat the state-of-art. The strain map created by py4DSTEM also has less precision in determining the strain values. When using CC-ST-AE, the generated strain map is clearer, and with higher image quality. More of the known strain features are apparent in the real-space image and strain histogram.

This model shows several significant improvements including: 1. When tested on simulated data, the model can generate ground truth with minimal error. 2. The model can learn the rotation and strain on noisy diffraction patterns where correlation strain mapping failed and significantly outperforms py4DSTEM. 3. Our model can accommodate large and continuous rotations difficult with other methods. 4. Our model is more robust to noisy data, especially on extremely noisy data where other strategies do not work. 5. Our model can map the strain, shear, and rotation; identify dislocation and ripples; and distinguish background and sample area automatically with improved robustness to noise.

## 3 Conclusion

Ultimately, this work demonstrates how embedding physical concepts into unsupervised neural networks can simplify, automate, and accelerate analysis pipelines while simultaneously leveraging stochastic averaging that improves the robustness of noisy data. This algorithmic concept can be extended to include other physical phenomena (e.g., polarization, sample tilt), can be used in automated experiments, and can be applied to other applications in materials characterization.

## References


[1] Colin Ophus. Four-dimensional scanning transmission electron microscopy (4d-stem): From scanning nanodiffraction to ptychography and beyond. Microscopy and Microanalysis, 25(3):563–582, 2019.

[2] Colin Ophus, Steven E Zeltmann, Alexandra Bruefach, Alexander Rakowski, Benjamin H Savitzky, Andrew M Minor, and Mary C Scott. Automated crystal orientation mapping in py4dstem using sparse correlation matching. Microscopy and microanalysis, 28(2):390–403, 2022.

[3] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. arXiv preprint arXiv:1511.08458, 2015.

[4] Alireza Makhzani and Brendan Frey. K-sparse autoencoders. arXiv preprint arXiv:1312.5663, 2013.

[5] Yimo Han, Kayla Nguyen, Michael Cao, Paul Cueva, Saien Xie, Mark W Tate, Prafull Purohit, Sol M Gruner, Jiwoong Park, and David A Muller. Strain mapping of two-dimensional heterostructures with subpicometer precision. Nano letters, 18(6):3746–3751, 2018.

[6] Joydeep Munshi, Alexander Rakowski, Benjamin H Savitzky, Steven E Zeltmann, Jim Ciston, Matthew Henderson, Shreyas Cholia, Andrew M Minor, Maria KY Chan, and Colin Ophus. Disentangling multiple scattering with deep learning: application to strain mapping from electron diffraction pat- terns. npj Computational Materials, 8(1):254, 2022.

[7] Nathanael P Kazmierczak, Madeline Van Winkle, Colin Ophus, Karen C Bustillo, Stephen Carr, Hamish G Brown, Jim Ciston, Takashi Taniguchi, Kenji Watanabe, and D Kwabena Bediako. Strain fields in twisted bilayer graphene. Nature materials, 20(7):956–963, 2021.

[8] SW Bedell, A Khakifirooz, and DK Sadana. Strain scaling for cmos. Mrs Bulletin, 39(2):131–137, 2014.

[9] Debangshu Mukherjee, Jocelyn TL Gamler, Sara E Skrabalak, and Raymond R Unocic. Lattice strain measurement of core@ shell electrocatalysts with 4d scanning transmission electron microscopy nanobeam electron diffraction. ACS Catalysis, 10(10):5529–5541, 2020.

[10] Thomas C Pekin, Christoph Gammer, Jim Ciston, Andrew M Minor, and Colin Ophus. Optimizing disk registration algorithms for nanobeam electron diffraction strain mapping. Ultramicroscopy, 176:170–176, 2017.

[11] Renliang Yuan, Jiong Zhang, and Jian-Min Zuo. Lattice strain mapping using circular though transform for electron diffraction disk detection. Ultramicroscopy, 207:112837, 2019.

[12] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. Advances in neural information processing systems, 28, 2015.