CS230

Classification of X-ray Single-Particle Images Using Multiple Output Convolutional Neural Networks

Xiangmeng Cai
Department of Bioengineering
Stanford University
xmcai@stanford.edu

Enci Liu
Department of Computer Science
Stanford University
jesslec@stanford.edu

Abstract

X-ray free-electron lasers single-particle imaging is a promising technology to determine the 3D structures of biological molecules. Due to the large amount of data collected during the experiments, automated data analysis is essential. Past research has shown that convolutional neural network (CNN) can be used to classify the diffraction patterns of the SPI experimental images. This study will present a novel multi-output model based on CNN that accomplishes diffraction pattern classification and particle type classification at the same time. Our experiment shows that both the 10-layer CNN and VGG-16 reach an accuracy above 90% on both tasks. Additionally, adjusting the weights of the two tasks in the loss function improves the performance of the CNN model. Our model provides a framework that can be extended in the future to generate more outputs. You can access the model here: `https://github.com/XiangmengCai/CS230_Project`

1 Introduction

X-ray free-electron lasers (XFELs) single-particle imaging (SPI) methods are used to generate high-throughput results in structural biology. During the experiment, X-ray pulses hit the particles into the radiation region and the diffraction patterns are recorded to reconstruct the 3D structures. Since its beginning, SPI experiment has been a main driver for structural biology discoveries, which has been used to determine the structures of viruses, bacteriophages, and organelles. [1, 2, 3]

The ideal experimental condition is when single particle interferes with the X-ray independently, so that the diffraction patterns of a single particle in random orientations can be collected by the detector and used to reconstruct the 3D structure. [4] However, XFEL pulses can intercept more than one particle, producing unusable diffraction patterns, which is known as the "multiple hits" problem, and the the task of classifying single and multiple-hits images is known as "pattern identification". [5]

Traditionally, these "multiple-hits" images are sorted manually, which is a time-consuming and error-prone process. Several programs have been developed for data processing, but they have limited ability in identifying multiple-hits images. [6, 7] In this project, we want to train a neural network for the task of pattern identification, which automatically filters out useful images after an X-ray imaging experiment. Additionally, we want to construct a novel multiple-output neural network to classify the images when there are multiple particles in the dataset, which is referred to as "particle classification" in this article. Our multi-output model will be flexible so that it can accommodate other tasks in the future, such as classifying images with different detector distance from the sample, or estimating resolution. Doing so will significantly improve the efficiency of XFEL experiments and help advance the field of structural biology.

CS230: Deep Learning, Spring 2021, Stanford University, CA.

2 Related Work

Deep learning approaches to identify multiple-hits images have gathered unprecedented advances in recent years. In their work published in 2019, Shi et al. successfully demonstrate that convolutional neural networks can be used to identify multiple-hits SPI images. Using simulated dataset, their CNN is able to achieve a 96% accuracy on classifying scattering patterns of single and multiple particles. [5]. Another work by Ignatenko et al. compared the performance of CNN, YOLOv2 and YOLOv3 on classifying multiple-hits images. Interestingly, they found that YOLOv2 performs significantly better than YOLOv3 on this task, and conclude that with a limited size of dataset and a comparatively simple task, a moderate size CNN, as well as YOLOv2, is better than a deeper network such as YOLOv3. [8]

All of these models deal with a single task of pattern identification. Our goal is to develop a multi-output model, which benefits from the fast-growing research in multi-output learning. In multi-output models, the goal is to simultaneously predict multiple outputs given a single input, which enables the model to solve complex decision-making problems. [9] In this work by Raunak Dey et al., the authors built a CNN for classifying lung cancer CT scans. Although the final output is a binary classification of benign/malignant, they discovered that by adding flatten layers that branch out from each pooling layer and feeding this intermediate output to the final layer, the accuracy of their model is significantly improved. [10] This inspires us as different tasks of classifying SPI images, whether number of hits, particle types, or detector distance, depend on the same images. Therefore, developing a single-input, multiple-output model will be more effective than developing a separate model for each task, as there can be shared features between tasks.

3 Dataset and Features

Considering the amount of samples we need to train the model on a diverse set of particles and the high error rate of human labeling, we need to use simulated SPI images that mimic the output from real world XFEL-SPI experiments. We used *Skopi* developed by Chun Hoon Yun et al. and *Cmtip* by Ariana Peck at SLAC to generate 77K simulated images on 11 different particles. [11, 12]

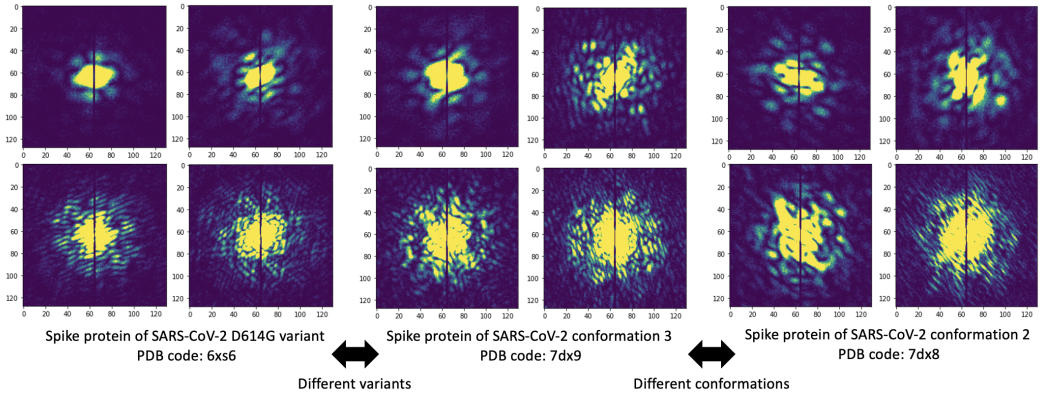


Figure 1: Examples of simulated datasets with single, double, triple, and quadruple particles. These three challenging datasets are structurally similar. Successfully classifying them will help with SPI experiments to reconstruct their 3D structures.

For each particle, we first downloaded the structure from Protein Data Bank. Then, we used the simulator that calculates the pixel intensities under four different hit conditions, i.e. single-, double-, triple-, and quadruple-hits. Since the main task is to distinguish single-hit images from multiple-hit images, we generated 4K images for single-hit, and 1K for each other categories. These images are monolithic and were downsampled to 128×128 pixels size. We included a diverse sample of particles, from bacteriophage to chaperonin protein. To add more complexity to our task, we included three datasets from SARS-CoV-2 spike protein. Specifically, we included two different conformations of the spike protein bound to the human ACE2, the target that grants SARS-CoV-2 infectiousness (PDB code: 7dx8, 7dx9). We also included structure 6xs6, which is the SARS-CoV-2 spike protein with

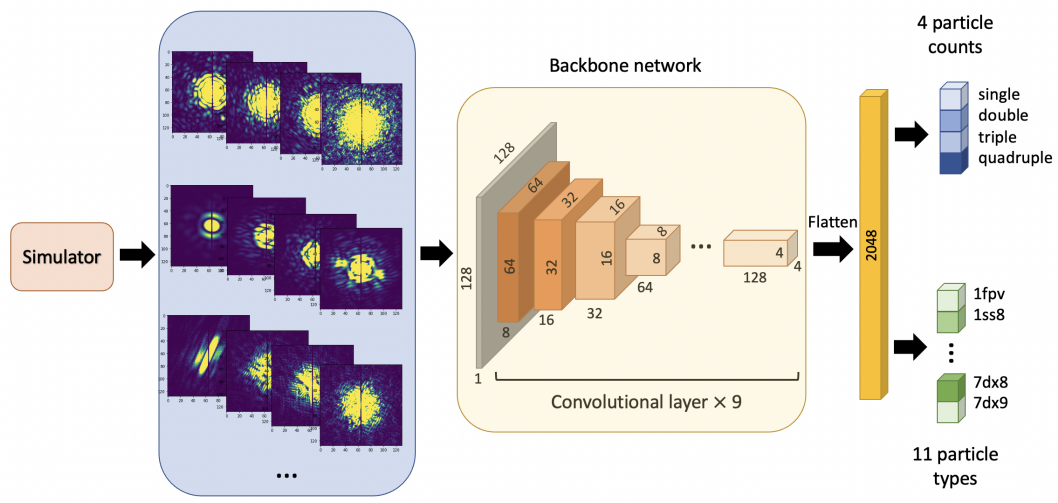


Figure 2: The architecture of the proposed multi-output classifier.

the D614G mutation. Classifying these images is challenging because these particles are structurally very similar.

4 Method

The goal is to predict two labels - particle type and count - given a single input X-ray image. For this purpose, we build an end-to-end multi-output classification pipeline comprised of two main parts. First, we use the simulator to generate specific types and counts of particle X-ray images (described in Section 3). The synthetic data is then used as input to the multi-output classifier, which includes the backbone network shared between two branches and the parallel fully-connected (FC) branches for the two output labels. The proposed pipeline is adaptable because the backbone network can be easily replaced with different model architectures like ResNet [13] and VGG [14].

4.1 Multi-Output CNN

The Multi-output CNN consists of 9 shared convolutional layers and separated parallel FC layers for each output. Given an input image of shape $1 \times 128 \times 128$, we first pass it through four convolutional layers with kernel size $K = 2$ and stride $S = 2$. We also increase the number of channels from 1 in the original input to 64 in the first four layers to allow the model to learn various features. In the next five convolutional layers, we use $K = 2$ and $S = 1$ to reduce the height and width more slowly. The shared layers allow the model to learn features that are important to both output labels that we are predicting. Parameter sharing also makes the proposed method memory and computationally efficient. After passing through all the 9 shared convolutional layers, we flatten the feature of shape $128 \times 4 \times 4$ and pass it to two parallel FC layers, each followed by a Softmax layer.

To decide the number of CNN layers, we also compare the performance of models with 3, 5, and 18 layers. However, we see that the 10-layer multi-output CNN outperforms other numbers of layers by a large margin. Having only 3 and 5 layers are not sufficient to learn features for both output labels. Moreover, 18 layers is too deep and could lead to overfitting for X-ray images, which have relatively fewer and less complicated features than normal images. The comparison results are shown in Table 2 of Appendix.

4.2 Weighted Loss Function

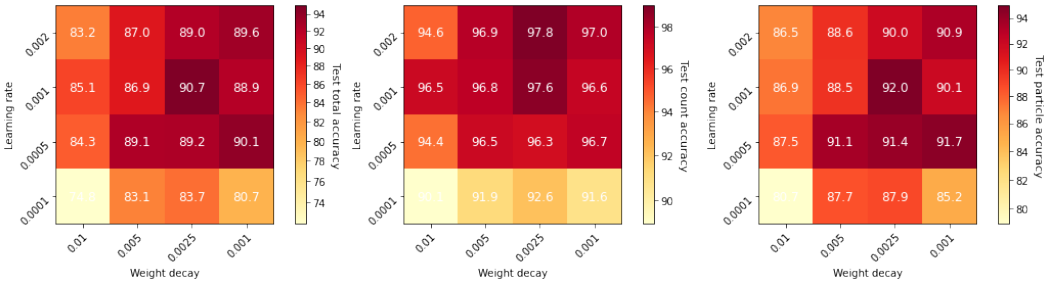
The two output labels might have different contribution and weight in feature learning. In particular, the model might find predicting one label more challenging and more feature-heavy than the other. To address this problem, we design a weighted loss function that gives different weights to the two loss terms when combining them.

First, we compute the loss terms for count label and particle label using cross entropy loss defined in PyTorch. Then, we give a larger weight of $\beta = 4$ to the loss term computed from the particle count label when summing the two terms. The final weighted loss is computed as:

$$\mathcal{L}_{total} = \beta \mathcal{L}_{count} + \mathcal{L}_{particle} \tag{1}$$

4.3 Hyperparameter Tuning

We explored choices of two hyperparameters: learning rate and weight decay. The hyperparameter combinations were evaluated on three test accuracies - total accuracy, count accuracy, and particle accuracy. We applied grid search to find the optimal combination of the two hyperparameters as shown in Figure 3. The final choice of hyperparameters are shown in Table 3 in Appendix.



(a) Test total accuracy (b) Test count accuracy (c) Test particle accuracy

Figure 3: The hyperparameter grid search result of the proposed multi-output model. Fig 3a, 3b, and 3c represents test total, count, and particle accuracies, respectively. In each case, combinations of learning rate and weight decay are tested and the best one is used in the proposed model.

5 Result and Discussion

5.1 Comparison Experiments

We compare the test performance of the proposed method with two baseline networks - ResNet18 [13] and VGG16 [14]. During experiments, both networks are inserted into the proposed multi-output classification pipeline as the backbone network. We only added the two parallel FC layers at the end of each structure.

As shown in Table 1, the proposed multi-output CNN outperforms both ResNet18 and VGG16. One reason why the multi-output ResNet18 is having very low test accuracy is that skip connections is not useful in the case of the particle diffraction X-ray images. The residual architecture of ResNet18 is designed to skips some layers and speed up training [13], which also results in less exploration of the feature space. Since there are already not a big amount of features to learn in the input images, skipping some layers might cause the model to overlook some important features.

Furthermore, the proposed method only has 165K parameters, which is much less than the 11M and 15M of the two baselines, respectively. This means that our method could achieve comparable and even better results with less computation power on the task of multi-output classification of particle diffraction X-ray images.

5.2 Ablation Studies

To verify the effectiveness of key architectural design choices, we also conducted ablation studies. Specifically, we compared the performance of the complete model with: 1) the multi-output CNN with fewer shared parameters, or the branch-off-early version, and 2) the multi-output CNN with an unweighted loss.

By comparing the performance of the branch-off-early and branch-off-late versions in Table 1, we see that having less shared parameters - i.e. branching off for each of the output labels earlier - hurts both the count and particle accuracies. This is because the particle type and count of an image depends on similar features in the input image. Moreover, using weighted loss also boosts performance.

Backbone Networks	Test Accuracies (%)			# Parameters
	Total	Count	Particle	
ResNet18	13.3	56.1	23.5	11,177,999
VGG16	92.0	98.2	92.7	15,089,871
10-layer CNN (branch off early)	88.5	95.0	91.9	298,631
10-layer CNN (unweighted loss)	91.1	97.1	92.5	164,967
10-layer CNN (branch off late + weighted loss)	**92.7**	**98.7**	**93.5**	164,967

Table 1: Test accuracies - total, count, and particle - of baseline models and the proposed methods

5.3 Performance and structural similarity

One goal of our project is to classify particles with similar structures. To test the performance of our model, we included 3 structural similar datasets (7dx8, 7dx9, 6xs6). We found that the model achieves near perfect classification accuracy on all datasets except 7dx8 and 7dx9. We may need to perform data augmentation to improve the performance.

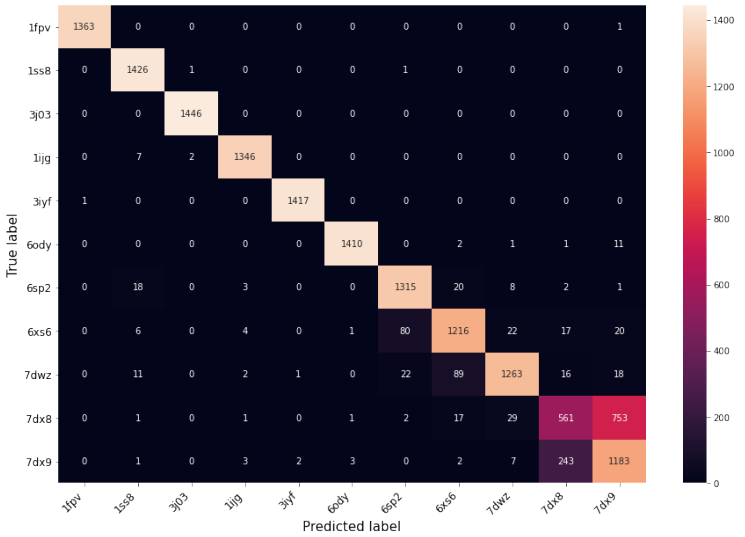


Figure 4: Confusion matrix of the particle classification. It shows that the model is able to correctly classify most particles, including 6xs6, but is less successful in distinguishing 7dx8 and 7dx9.

6 Conclusion and Future Work

In this study, we presents a single-input multiple-output CNN model to classify SPI images. To our knowledge, there is no previous work to apply multi-output networks on SPI images, so our model contributes to the field by providing an architecture that can be easily extended to classify other features of the SPI images. Additionally, our work shows that a medium size CNN performs better on SPI images than deeper networks such as ResNet18. We also show that a late-branching CNN is performs better than an early-branching one, probably due to that there are shared parameters between different tasks. This shows that a single-input multiple-output model is not only computationally efficient, but also performs better than training a separate model for each task. In future work, we hope to test our model on real world experiments. We will test the generalizabily of the model by testing it on particles it did not see before. We will also test the performance of the model on images with noise. Finally, we want to test the flexibility of the model by adding more outputs, such as estimating the resolution of the final 3D model based on the diffraction patterns. These experiments will help us deploy the model in SPI experiments and help advance the field of structural biology.

7 Contribution

Xiangmeng Cai generated simulated datasets, wrote code for loading VGG16 and ResNet18, and ran experiments. Enci Liu wrote code for dataloaders and CNN, and ran experiments. The paper is the common effort of the two authors.

We want to thank Chunhong Yoon, Ariana Peck, and Frédéric Poitevin fom Stanfod SLAC for their help with the project.

References

[1] Stephan Kassemeyer, Jan Steinbrener, and Lukas Lomb et al. Femtosecond free-electron laser x-ray diffraction data sets for algorithm development. *Opt. Express*, 20(4):4149–4158, Feb 2012.

[2] M. Marvin Seibert, Tomas Ekeberg, and Filipe R. N. C. et al. Maia. Single mimivirus particles intercepted and imaged with an X-ray laser. *Nature*, 470(7332):78–81, 2011.

[3] Max F. Hantke, Dirk Hasse, and Filipe R. N. C. et al. Maia. High-throughput imaging of heterogeneous cell organelles with an X-ray laser. *Nature Photonics*, 8(12):943–949, 2014.

[4] Hemanth K N Reddy, Chun Hong Yoon, Andrew Aquila, Awelet al.J, and Lourdu Xavier P. Coherent soft X-ray diffraction imaging of coliphage PR772 at the Linac coherent light source. *Scientific Data*, 4(1):170079, 2017.

[5] Y. Shi, K. Yin, X. Tai, H. DeMirci, A. Hosseinizadeh, B.G. Hogue, H. Li, A. Ourmazd, P. Schwander, I.A. Vartanyants, C.H. Yoon, A. Aquila, and H. Liu. Evaluation of the performance of classification algorithms for XFEL single-particle imaging data. *IUCrJ*, 6(2):331–340, 2019.

[6] D. Damiani, M. Dubrovin, I. Gaponenko, W. Kroeger, T.J. Lane, A. Mitra, C.P. O'Grady, A. Salnikov, A. Sanchez-Gonzalez, D. Schneider, and C.H. Yoon. Linac Coherent Light Source data analysis using psana. *Journal of Applied Crystallography*, 49(2):672–679, 2016.

[7] A. Barty, R.A. Kirian, F.R.N.C. Maia, M. Hantke, C.H. Yoon, T.A. White, and H. Chapman. Cheetah: software for high-throughput reduction and analysis of serial femtosecond X-ray diffraction data. *Journal of Applied Crystallography*, 47(3):1118–1131, 2014.

[8] Alexandr Ignatenko, Dameli Assalauova, Sergey A Bobkov, Luca Gelisio, Anton B Teslyuk, Viacheslav A Ilyin, and Ivan A Vartanyants. Classification of diffraction patterns in single particle imaging experiments performed at x-ray free-electron lasers using a convolutional neural network. *Machine Learning: Science and Technology*, 2(2):025014, Feb 2021.

[9] Donna Xu, Yaxin Shi, Ivor W Tsang, Yew-Soon Ong, Chen Gong, and Xiaobo Shen. A Survey on Multi-output Learning. *arXiv*, 2019.

[10] Raunak Dey, Zhongjie Lu, and Yi Hong. Diagnostic Classification of Lung Nodules Using 3D Neural Networks. *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pages 774–778, 2018.

[11] Chunhong Yoon. Skopi: Single particle imaging simulation package. `https://github.com/chuckie82/skopi`, 2021.

[12] Ariana Peck. Cmtip: Cartesian mtip (multitiered iterative phasing). `https://github.com/apeck12/cmtip`, 2021.

[13] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[14] K. Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2015.

8 Appendix

8.1 Dataset Generation

The structures we used to generate simulated images were downloaded from the Protein Data Bank. 11 such structures were used for simulation and their PDB codes are: 1fpv, 1ijg, 1ss8, 3iyf, 3j03, 6ody, 6sp2, 6xs6, 7dwz, 7dx8, 7dx9.

8.2 Results from various CNN architectures

To find the optimal CNN model, we tested with different scales of the CNN models. We found that the 10-layer CNN model has the best performance. This agrees with past work showing that for SPI images, a medium size CNN works better than deeper networks. Therefore, we choose the 10-layer CNN as the proposed model for the task.

# Layers	Test accuracy (%)			# Parameters
	Total	Count	Particle	
3-layer	85.2	91.5	92.4	100,567
5-layer	88.6	94.5	93.0	74,535
10-layer	91.2	97.1	92.5	164,967
18-layer	83.4	95.0	85.6	204,167

Table 2: Number of layers' influence on test performances.

8.3 Different branching architectures

In Table. 1 we showed that the late-branching CNN performs better than the early-branching one. We hypothesize that this is because our two tasks – identifying the number and type of particles – have shared features. Therefore, the late-branching model, with more share layers, has better performance.

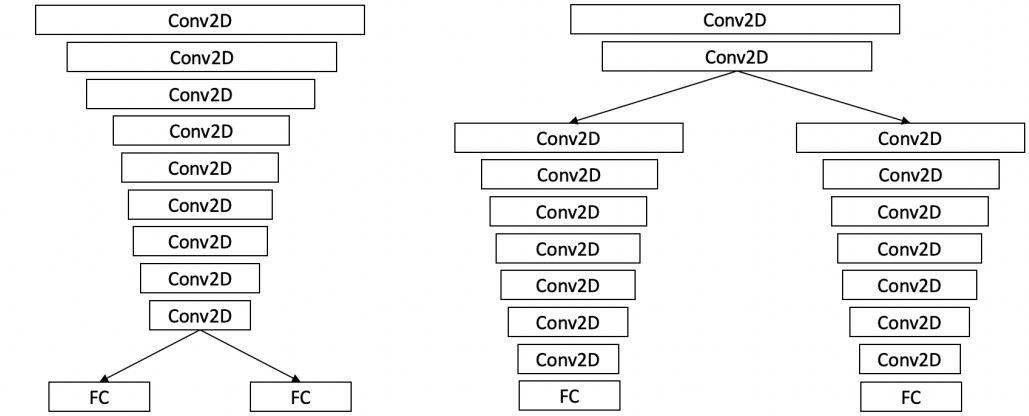


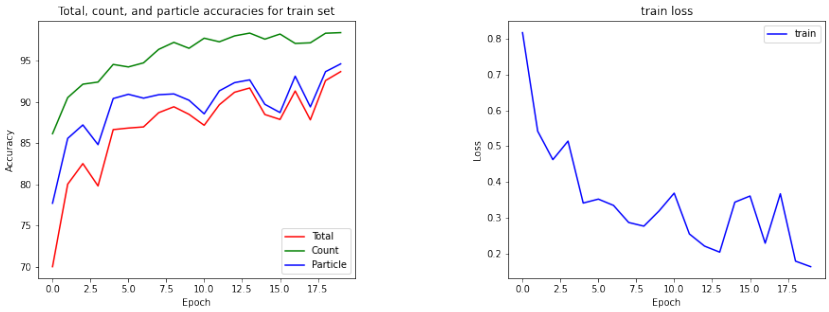
Figure 5: The branch-off-late and the branch-off-early version of the multi-output CNN.

8.4 Hyperparameters used in the proposed model

Hyperparameter Choices
learning rate = 0.001
weight decay = 0.001
layers (N) = 10
loss weight (β) = 4

Table 3: Hyperparameter choices

8.5 Results from proposed model



(a) Train accuracies

(b) Train loss

Figure 6: The train accuracies and loss for the proposed method (10-layer CNN). The model is trained for 20 epochs.

8.6 Error analysis with 7dx8/7dx9

Here are some examples of mis-labeled 7dx8 images. We included 7dx8 and 7dx9, two highly similar particles, to test the performance of the network when classifying similar objects. Here we found that most of the mis-labeled 7dx8 were classified as 7dx9, which agrees with our concern that our model may not work well when structures are extremely similar.

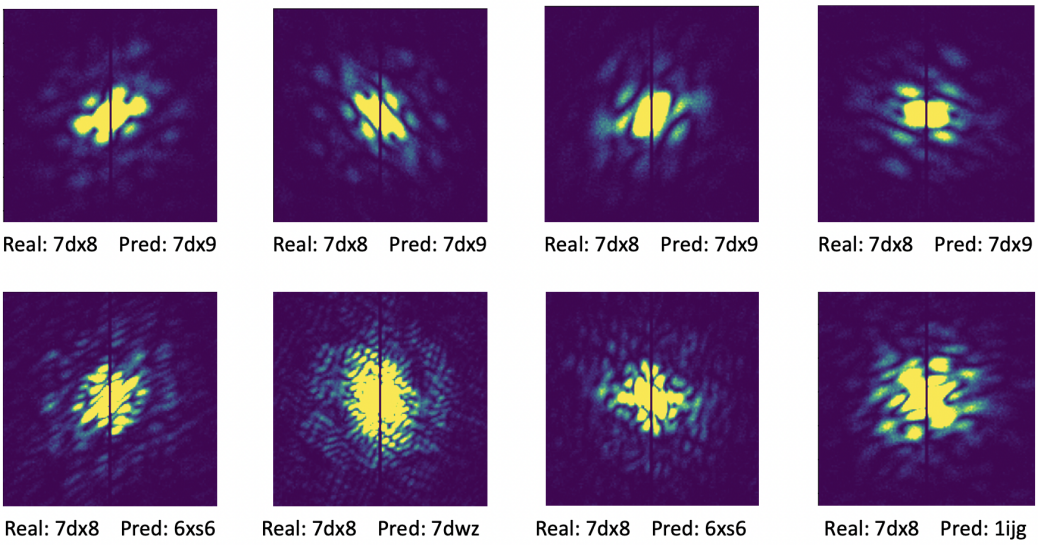


Figure 7: Wrong predictions of particle *7dx8*.