DIFFRANET: AUTOMATIC CLASSIFICATION OF SERIAL CRYSTALLOGRAPHY DIFFRACTION PATTERNS

Anonymous authors
Paper under double-blind review

ABSTRACT

Serial crystallography is the field of science that studies the structure and properties of crystals via diffraction patterns. In this paper, we introduce a new serial crystallography dataset generated through the use of a simulator; the synthetic images are labeled and they are both scalable and accurate. The resulting synthetic dataset is called DiffraNet, and it is composed of 25,000 512x512 grayscale labeled images. We explore several computer vision approaches for classification on DiffraNet such as standard feature extraction algorithms associated with Random Forests and Support Vector Machines but also an end-to-end CNN topology dubbed DeepFreak tailored to work on this new dataset. All implementations are publicly available and have been fine-tuned using off-the-shelf AutoML optimization tools for a fair comparison. Our best model achieves 98.5% accuracy. We believe that the DiffraNet dataset and its classification methods will have in the long term a positive impact in accelerating discoveries in many disciplines, including chemistry, geology, biology, materials science, metallurgy, and physics.

1 INTRODUCTION

Real-time feedback on diffraction images is vital in Crystallography (Berntson et al. (2003); Ke et al. (2018)). Crystallography (Woolfson (1997)) is the science that studies properties of crystals. It makes use of X-ray diffraction to infer structures of crystals. Broadly, a crystal is irradiated with an X-ray beam that strikes the crystal and produces an image with the diffraction pattern (Fig. 1). Images are captured by a detector that runs at 130 Hz. At present serial crystallography, scientists have to screen tons of images by manual classification. This process is not only error-prone but also has the effect of slowing down the overall discovery process.

In this paper, we introduce a method for generating labeled diffraction images. The technique produces and labels images via a simulator and, therefore, the process is both scalable and accurate. The simulator receives as input the properties of the incident X-ray beam, the environment, and the structure to be analyzed and generates synthetic diffraction images. Since the process is simulated and controlled, the dataset annotation is 100% accurate, an impossible feat for manually annotated real images.

As a result of the simulator we introduce DiffraNet, the first dataset of serial crystallography diffraction synthetic images. DiffraNet is composed of 25,000 512x512 grayscale labeled images and we open it to the rest of the community. DiffraNet is comprised of five classes, each representing a possible outcome of the serial crystallography experiment. Of the five possible classes, two classes denote images with no diffraction patterns (an undesired outcome) and the other three denote images with varying degrees of diffraction. DiffraNet contains fundamentally different images with respect to standard image datasets such as ImageNet (Deng et al. (2009)) and the CIFARs (Krizhevsky (1993)), see Fig. 2.

Finally, we also present three different approaches for classifying diffraction images. First, a method based on a mix of feature extractors and Random Forests (RF). Second, a combination of feature extractors and Support Vector Machines (SVM). Last, DeepFreak, a Convolutional Neural Network (CNN) topology based on ResNet-50. All approaches are open-source and have been fine-tuned using AutoML hyperparameter optimization tools such as Hyperopt (Bergstra et al. (2013)) and BOHB (Falkner et al. (2018)). These approaches achieve 98.45%, 97.66%, 98.5% accuracy, respectively.

To sum up, the contributions of this paper are:

- A new, openly available, classification dataset dubbed *DiffraNet* for image diffraction in the serial crystallography experimental setting.
- Three classification methods based on RFs, SVMs and CNNs able to classify the diffraction images with up to 98.5% accuracy.
- An open-source implementation of the newly introduced approaches.

The rest of this paper is organized as follows. Section 2 presents a background in X-ray crystallography and a summary of related work. Section 3 describes our simulator and the DiffraNet dataset. Sections 4 and 5 describe our approaches for classifying diffraction images from DiffraNet and the experimental results achieved. Finally, Section 6 concludes this work and presents future work.

2 BACKGROUND

2.1 CRYSTALLOGRAPHY

Serial Crystallography (Stellato et al. (2014)) refers to a more recent crystallography technique for investigating properties from hundreds of thousands of microcrystals using X-ray free-electron laser. It is used in many disciplines, including chemistry, geology, biology, materials science, metallurgy, and physics. It has been a central tool in driving significant increases in understanding processes from solid-state physics to molecular biology to synthetic chemistry (Woolfson (1997)). This understanding, in turn, has led to substantial advances in, for instance, drugs development for fighting diseases.

Crystallography makes use of X-ray diffraction to infer the structure of crystal samples (Woolfson (1997)). First, a crystal is irradiated with an X-ray beam. As X-ray photons strike the crystal, some will diffract due to the geometry of the lattice and produce a diffraction pattern unique to the material as in Fig. 1. These patterns are recorded by a detector (usually phosphor or silicon) and make it possible to infer information about the crystal, like the chemical bonds and disorder of its atoms.

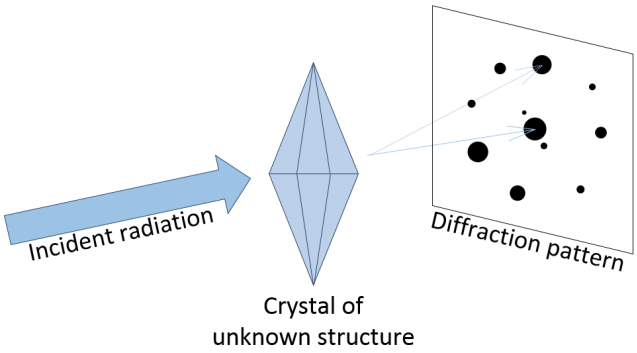

Figure 1: Generic scheme depicting a crystallography experiment.

Analysis and feedback on diffraction images are paramount in both conventional and serial crystallography (Berntson et al. (2003); Ke et al. (2018)). Recent technological advances have automated and accelerated crystallography experiment steps and, in turn, allowed researchers to generate diffraction results at unprecedented speeds. However, as no system currently exists to provide real-time analysis of the diffraction images produced, many of the compelling advantages afforded by these technological leaps cannot be fully utilized. Besides, without timely feedback, expensive and limited quantity samples may be wasted because of problems regarding experimental optimization, sample positioning, or X-ray beam alignment. This paper addresses the automation of serial crystallography images screening.

2.2 PREVIOUS APPROACHES TO CLASSIFICATION ON SERIAL CRYSTALLOGRAPHY

Several studies have been done for trying to automatically classify images derived from crystallography phenomena (Berntson et al. (2003); Becker & Streit (2014); Yann & Tang (2016); Park et al. (2017); Bruno et al. (2018); Ke et al. (2018); Ziletti et al. (2018)).

In particular, Bruno et al. (2018) employed CNNs for classifying outcomes of crystallization processes. The model they used is a variation of *Inception-v3* (Szegedy et al. (2016)), images were categorized in the following four classes: clear, precipitate, crystal, and other. The dataset used in this study has nearly half a million images, and around 10% of them were used for testing. They achieved 94% accuracy on the test set, approximately.

Ziletti et al. (2018) used CNNs to classify crystal structures, i.e., the way atoms inside a crystal are arranged. By using diffraction images, they were able to represent and classify a dataset with around 100,000 crystal structures. Park et al. (2017) worked on classifying powder X-ray diffraction patterns using CNNs achieving 94.99% of accuracy. Yann & Tang (2016) aimed at analyzing protein crystallization-trial images. Notably, their CNN approach dubbed *CrystalNet* hits around 8% and 20% improvement in overall accuracy compared to the Random Forests and Nearest Neighbor approaches, respectively.

Similar to our work, Ke et al. (2018) used a CNN for detecting Bragg spots on crystallography diffraction images. Their CNN employs a structure similar to that of *AlexNet* (Krizhevsky et al. (2012)) and comprises four sets of layers: convolution, batch normalization, rectification, and downsampling (max pooling). They used local contrast normalization to enhance the contrast between background and Bragg spots. They also augmented the dataset through the use of random and center cropping. Ke *et al.* used a human expert annotated dataset, consisting of 2,000 images, as the ground truth and compared their CNN accuracy against with automatic spot-finding tools. They achieved around 93% accuracy in classifying images as a *hit*, *maybe*, or *miss*. Respectively, these classes refer to when an image does, might, and does not possess Bragg spots.

Our work sets apart from these above in several ways. First, our process of labeling data is scalable and accurate. Second, our dataset is tailored to a specialized application: Crystallography. Third, we explore different computer vision techniques for classification. Last, we use multiple AutoML optimization tools to achieve the best results in each setting.

2.3 IMAGE DATASETS

Today, there is a great deal of publicly available datasets for training machine learning models. Few notorious datasets are: *ImageNet* (Deng et al. (2009)), *CIFAR-10/100* (Krizhevsky (1993)) and *COCO* (Lin et al. (2014)). ImageNet, for example, comprises around 14 million images following the WordNet hierarchy organization. On average, each node of the ImageNet's hierarchy has 500 images. Some popular ImageNet synsets include *animal, plant, material, and activity*. Common Objects in Context, or COCO for short, is an annotated dataset consisting of images portraying scenes from everyday life and their ordinary objects. COCO features, for instance, 200,000 labeled images, 1.5 million object instances, and 250,000 people with keypoints. The CIFAR-10 and CIFAR-100 are annotated samples of the Tiny Images Dataset (Krizhevsky (1993)). CIFAR-10 comprises 60,000 images divided into ten (airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck) classes containing 6,000 images each. Of these, 50,000 are for training and the rest for testing. Its larger counterpart, CIFAR-100, is much like CIFAR-10, but it is made up of 100 classes with 600 images each.

3 THE DIFFRANET DATASET

We introduce a new, openly available, synthetic dataset of diffraction images dubbed *DiffraNet*. Experimental diffraction images are difficult to classify on a large scale. Highly trained experts are needed to categorize these images manually, and the process is both slow and error-prone. Thus, we synthetically generated our dataset. DiffraNet is 100% accurate because labels derive images, not the other way round.

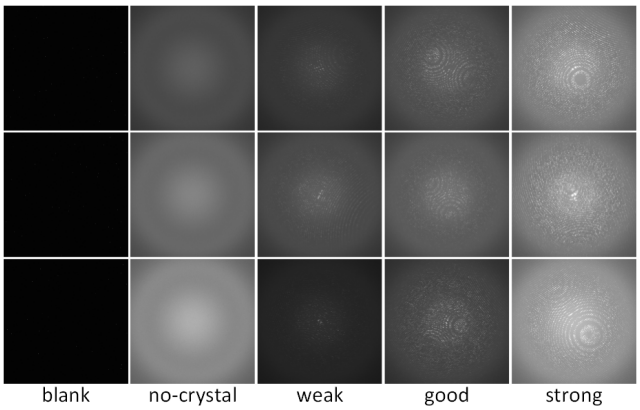

Figure 2: DiffraNet dataset classes.

DiffraNet was generated using the nanoBragg simulator[1]. The physics of X-ray diffraction are well understood and we have developed simulators for the entire process of producing diffraction images. The input to nanoBragg includes X-ray beam properties (flux, beam size, divergence, and bandpass), crystal properties (unit cell, number of cells, and a structure factor table), and the experimental parameters (sources of background noise, detector point-spread, and shadows – such as the beamstop). The simulation is computationally intensive but highly parallelizable: images can be rendered and labeled at an average rate of one per second on the 384-core SMB cluster.

DiffraNet comprises 25,000 512x512 grayscale images. The classes are *blank*, *no-crystal*, *weak*, *good*, or *strong*. *Blank* denotes an image with no X-rays and only detector noise while *No-crystal* the diffraction from amorphous carrier material but no crystalline matter. *Weak*, *Good*, and *Strong*, in turn, denote images with a crystal in the beam with increasingly stronger contribution to the pattern: *Weak* has small or faint diffraction patterns, *Good* has slightly larger and more discernible patterns, and *Strong* are ideal images, with large and clear diffraction patterns (Fig. 2).

DiffraNet is publicly available[2] and can be used for training, validating, and testing machine learning models. The primary goal in the classification of DiffraNet is to differentiate between classes with and without crystal diffraction patterns so that images without diffraction pattern can be discarded and downstream analysis can focus on images that are the most promising. DiffraNet partitions the dataset into training (40% of the dataset with a total of 10,000 images), validation (9.6% of the dataset with a total of 2,400 images), and test sets (50.4% of the dataset with a total of 12,600 images).

4 Classification on Image Diffraction

We propose three approaches for the classification of the DiffraNet dataset introduced in 3. The first two approaches rely on RFs and SVMs combined with feature extractors and the third approach is a CNN. We use off-the-shelf AutoML tools to search the hyperparameter space of the three classifiers automatically. We adopt two different tools: Hyperopt (Bergstra et al. (2013)) for the RF and SVM classifiers, and BOHB (Falkner et al. (2018)) for the CNN classifier. We use BOHB for the CNN because it includes a multi-fidelity feature that accelerates searches on CNNs. Both tools are based on Tree Parzen Estimator (TPE) (Bergstra et al. (2011)) models.

4.1 Feature Extractors

We implement three feature extractors to use together with our RF and SVM classifiers. Specifically, we use the Scale Invariant Feature Transform (SIFT, Lowe (2004)) with the Bag-of-Visual-Words approach (BoVW, Yang et al. (2007)) as local feature extractor, and the Gray-level Co-occurrence

[1] `http://doubleblind.com`
[2] `http://doubleblind.com`

Matrix (GLCM, Haralick et al. (1973)) and Local Binary Patterns (LBP, Ojala et al. (2002)) as global feature extractors. We choose these extractors because of their strong performance in image classification tasks (Kumar et al. (2017)) and in particular GLCM and LBP for their global texture features that are suitable in describing the images in DiffraNet.

We implement the feature extractors in Python using the OpenCV and scikit-image libraries. Also, we fine-tune the parameters of the extractors and both the SVM and RF classifiers using the Hyperopt Python library (Bergstra et al. (2013)). For the SIFT + BoVW extractor, we use the k-means algorithm to aggregate the visual codewords and optimize the size of the codebook by tuning the number of clusters in the k-means algorithm. For GLCM, we tune the distances and angles between pixel value pairs and use six Haralick features (Haralick et al. (1973)): Contrast, Dissimilarity, Homogeneity, Angular Second Moment, Energy, and Correlation. Finally, for LBP we tune the radius and number of points parameters that define the neighborhood size used by the extractor to compute the binary patterns. The feature extractors search space is summarized in Appendix A.

4.2 RF AND SVM CLASSIFIERS

RFs (Breiman (2001); Criminisi et al. (2012)) is an ensemble learning technique that can be used for both classification and regression. RFs create a forest of *decision trees*, a supervised learning technique for decision-making processes. A *randomized decision tree*, in turn, randomly select attributes out of a set of randomly chosen training samples.

SVMs (Vapnik (1995)) is a supervised learning technique used for classification and regression. For (almost) linearly separable data SVMs are straightforward: given labeled training data, SVMs output a separating hyperplane which may be then used to classify unlabeled data. For data that is not linearly separable, on the other hand, SVMs first employ *kernel functions* to map that onto another—often higher—dimensional space where the data is (almost) linearly separable and then, accordingly, proceed by finding a hyperplane.

We use Hyperopt to search for feature extractors and classifier hyperparameters jointly. Hyperopt proceeds by choosing one feature extractor and one classifier and then choosing a hyperparameter configuration based on the search spaces of each. By optimizing the extractor and classifier together, we allow Hyperopt to estimate particular extractor and classifier combinations that function well together. The search space is summarized in Appendix A.

4.3 THE DEEPFREAK NEURAL NETWORK

We introduce a new CNN dubbed *DeepFreak*. DeepFreak uses an adapted version of the Residual Neural Network with 50 layers (ResNet-50, He et al. (2016)). ResNet introduces *identity shortcut connections* that bypass one or more layers (as in Highway Networks, Srivastava et al. (2015)) with the addition of residual blocks which let the stacked layers fit a residual mapping instead of directly fitting the desired underlying mapping. This helps to address the vanishing gradient problem of deep networks.

We use PyTorch's official implementation of ResNet-50 as a baseline (PyTorch (2018)), which, by default, presumes images of size 224x224 from the ImageNet dataset as input, as opposed to our 512x512 DiffraNet images. We design a set of potential adjustments to the official implementation and use BOHB (Falkner et al. (2018)) to find the best topology and hyperparameter combination; the reader can refer to Appendix B for more details on the DeepFreak search space. BOHB is an AutoML tool based on Hyperband (Li et al. (2017)) and Bayesian Optimization (Bergstra et al. (2011)). It uses an iterative algorithm parameterized by two hyperparameters: maximum budget and η. These hyperparameters define how many configurations are evaluated per iteration and for how many epochs the network uses each configuration. In every iteration, BOHB assigns a budget—equal to or lower than the maximum budget—to all the configurations sampled. For each iteration i, BOHB keeps $1/\eta$ of the configurations tested in the iteration $i-1$ and increase the budget assigned to each configuration, up to the maximum budget. Our ultimate goal in using BOHB is to downsample the network so that DeepFreak trains faster and achieves higher accuracy.

We run BOHB on DeepFreak with a maximum budget of 50 epochs and $\eta = 3$. The best topology and hyperparameters found extends the strides of ResNet-50's last two blocks to 3 (instead of 2), uses a batch size of 1, a weight decay of 3.4855×10^{-5}, and a momentum of 0.56168. The learning

Table 1: DeepFreak topology hyperparameters.

Hyperparameter	**Value**
Number of filters	64
1st convolution kernel	7
1st convolution stride	2
1st pool size	3
1st pool stride	2
2nd pool size	7
2nd pool stride	1
Number of blocks	3, 4, 6, 3
Block strides	1, 2, 3, 3

Table 2: DeepFreak learning hyperparameters.

Hyperparameter	**Value**
Number of epochs	180
Optimizer	SGD
Learning rate	8.4474×10^{-4}
Decay epochs	Every 10 epochs
Decay rate	0.1
Momentum	0.56168
Weight decay	3.4855×10^{-5}
Loss function	Cross-Entropy
Batch size	1

Table 3: Best configuration of RF (left) and SVM (right) and accuracy on the validation set.

Hyperparameter	Values
GLCM Distances	[1, 2, 5, 8]
GLCM Angles	[45, 135]
Max Depth	20
Max Features	$\sqrt{features}$
Number of Trees	100
Class Weights	None
Accuracy	98.58%

Hyperparameter	Values
GLCM Distances	[1, 5, 8]
GLCM Angles	[0, 90, 135]
C	32
γ	0.5
Class Weights	[0.25, 0.25, 0.166, 0.166, 0.166]
Accuracy	97.88%

rate starts at 8.4474×10^{-4} and decays by 10 every 10 epochs. We split DiffraNet into training, validation, and testing (c.f. Section 3) and train the network for over 180 epochs. For each image in the training set, we rescale the pixel values to the [0, 1] range and subtract the per-pixel mean. DeepFreak configuration is summarized in Tables 1 and 2 and our code has been made publicly available (*authors omitted* (2018)).

5 Experiments

5.1 Hyperopt Results

We present the best configurations found by Hyperopt for the SVM and RF classifiers and their performance in the validation set. For this experiment, we have run Hyperopt for 150 iterations on a machine with 2 Intel Xeon E5 processors. The optimization has taken roughly 36 hours to complete, and the results are shown in Table 3.

RF and SVM classifiers have achieved 98.58% and 97.88%, respectively, i.e., RF has performed slightly better than SVM (0.7%). Note that the highest accuracy for both SVM and RF use GLCM as a feature extractor. This accuracy indicates that the GLCM works better than LBP and SIFT in the DiffraNet dataset. Precisely, GLCM has been the best extractor (98.58% accuracy), with LBP closely behind (96.71% accuracy). These results corroborate our hypothesis that global texture extractors would fit DiffraNet better. On the other hand, the SIFT + BoVW extractor has achieved 56.4% accuracy. This low accuracy is not surprising since SIFT looks for features in corners and objects of images, which are unusual in images from DiffraNet. Table 4 exhibits the best results achieved by classifiers for each feature extractor.

5.2 BOHB Results

We present the results of the DeepFreak optimization using BOHB. Here, we have run BOHB for 16 iterations in parallel in a machine with 2 Nvidia GeForce GTX 1080 Ti GPUs. The optimization has taken about nine days.

We have run the three best configurations (A, B, and C) found by BOHB, five times each. Figure 3 shows the mean learning curve (thin line) and the 80% confidence interval (shade) for each configu-

Table 4: Feature extractors and models best accuracy on validation set.

Feature Extractors	Model	Hyperparameters	Values	Validation Accuracy
GLCM	RF	Distances Angles	[1, 2, 5 8] [45, 135]	98.58%
LBP	SVM	Points Radius	24 3	96.71%
SIFT	RF	Clusters	25	56.42%

ration. Note A and B have highest mean accuracy; B has a broader confidence interval. We believe A has a narrower confidence interval due to its larger pooling layer, which leads to faster downsampling. Given the variability of CNNs training, we use these curves to choose the best DeepFreak configuration by analyzing mean and variance of these configurations. We have chosen B due to its highest validation accuracy over the networks we trained. The details on the three configurations used in this experiment are shown in Appendix C.

The final learning curve for DeepFreak, with the best configuration found by BOHB, is shown in Figure 4; the details on this configuration have been discussed in Section 4.3. We have trained the network for 180 epochs; the training converged after around 20 epochs. After training, DeepFreak achieved 98.42% validation accuracy, a result similar to RF and superior to SVM.

5.3 Test Set Results

Last, we have run the best configuration for our three classifiers over DiffraNet's test set, results are shown in Table 5. Note that the results are similar to those in the validation set, indicating that the three models can generalize the training data. Besides, all classifiers achieved over 97.6% accuracy. Notably, DeepFreak achieved the highest accuracy. Precisely, the accuracy of DeepFreak on the test set has been higher than that on the validation set, surpassing the RF and SVM by 0.06% and 0.85%, respectively.

We show the DeepFreak confusion matrix on the test set in Table 6; we show the RF and SVM confusion matrices on the test set in Appendix D. Note that misclassification often happens between *weak* and *good* and between *good* and *strong*. This behavior is natural since classes in these pairs are similar. Besides, note that DeepFreak hits near perfect results on the *blank* and *noxtal* classes, as evidenced by their precision (99.95% and 99.45%) and recall values (100% and 99.94%). This result means we can discard images without diffraction patterns with 99.83% accuracy. This is an

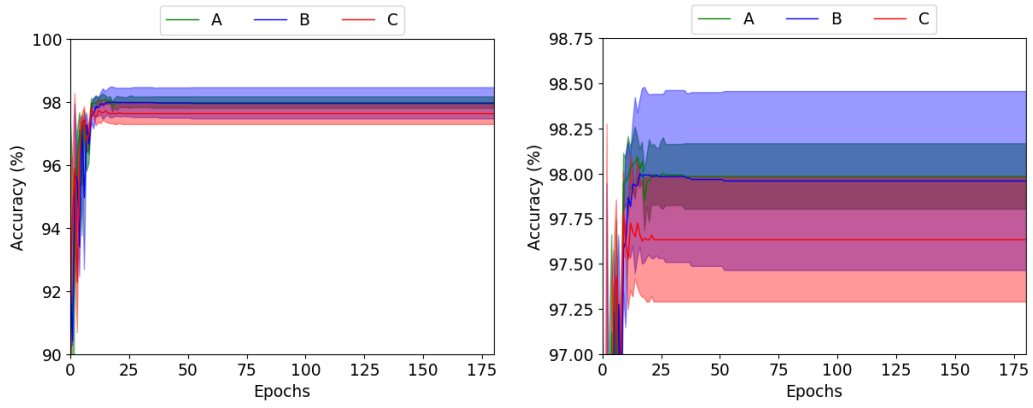

Figure 3: Mean accuracy (thin line) and 80% confidence interval (shade) of the three best configurations found by BOHB on the validation set. Each configuration was run five times for 180 epochs.

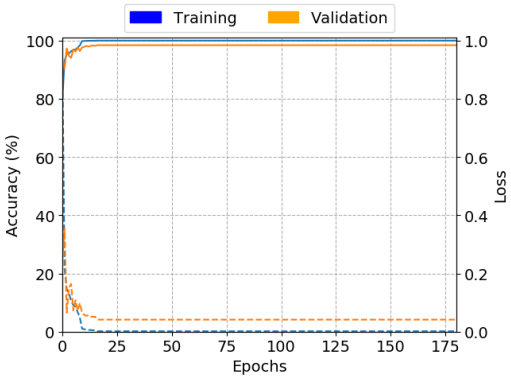

Figure 4: DeepFreak accuracy (solid line) and loss (dashed line) curves on training and validation sets; 180 epochs in total.

Feature Extractor	Classifier	Accuracy
GLCM	RF	98.45%
GLCM	SVM	97.66%
n/a	DeepFreak	**98.51%**

Table 5: DeepFreak, RF, and SVM accuracies on DiffraNet's test set.

important result for this application domain because it is important not to discard useful images; as mentioned in Section 3, it is less problematic to misclassify between the classes *weak, good, strong*.

Table 6: DeepFreak confusion matrix for the test set.

		Predicted class					
		blank	noxtal	weak	good	strong	Recall (%)
True class	blank	2069	0	0	0	0	100
	noxtal	0	3266	2	0	0	99.94
	weak	1	18	3280	47	0	98.03
	good	0	0	38	2368	38	96.89
	strong	0	0	0	44	1428	97.01
	Precision (%)	99.95	99.45	98.80	96.3	97.41	

6 Conclusions and Future Work

We have tackled the challenge of real-time classification of serial crystallography diffraction images. We have developed a method for generating accurately labeled synthetic diffraction images and used that to generate DiffraNet. DiffraNet comprises 25,000 512x512 grayscale diffraction images, each tagged as one out of five classes representing possible outcomes from crystallography experiments. DiffraNet is publicly available and can be used for training, validating, and testing machine learning models tailored to crystallography.

We have also explored several computer vision classification approaches. They are based on a blend of standard feature extractors with the RF and SVM classifiers and on an end-to-end CNN architecture called DeepFreak. All of our approaches have been fine-tuned with AutoML tools and tested over DiffraNet. Our results show that DeepFreak obtained the highest accuracy (98.51%), but we note that all of our classifiers hit over 97.6% accuracy. Moreover, DeepFreak achieved 99.83% accuracy in distinguishing between images with and without diffraction patterns.

In future iterations of the DiffraNet dataset we plan to add new images and new classes that are common place in serial crystallography. As an example, a class that is valuable in practice is to detect the presence of ice in the images. Images with ice indicate problems with the experiment setup that can disrupt the results and even damage the detector. It is important to detect and address these problems in a real-time feedback loop.

REFERENCES

Daniel Becker and Achim Streit. A neural network based pre-selection of big data in photon science. In *International Conference on Big Data and Cloud Computing (BdCloud)*, 2014.

James Bergstra, Daniel Yamins, and David Daniel Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International Conference on Machine Learning (ICML)*, 2013.

James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *International Conference on Neural Information Processing Systems (NIPS)*, 2011.

Andrea Berntson, Vivian Stojanoff, and Hiroshi Takai. Application of a neural network in high-throughput protein crystallography. *Journal of synchrotron radiation*, 10(6):445–449, 2003.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Andrew E Bruno, Patrick Charbonneau, Janet Newman, Edward H Snell, David R So, Vincent Vanhoucke, Christopher J Watkins, Shawn Williams, and Julie Wilson. Classification of crystallization outcomes using deep convolutional neural networks. *PLOS one*, 13(6):e0198883, 2018.

Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends® in Computer Graphics and Vision*, 7(2–3):81–227, 2012.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Stefan Falkner, Aaron Klein, and Frank Hutter. Bohb: Robust and efficient hyperparameter optimization at scale. *arXiv preprint arXiv:1807.01774*, 2018.

Robert M. Haralick, Karthikeyan Shanmugam, and Its'Hak Dinstein. Textural features for image classification. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-3(6):610–621, 1973.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

Tsung-Wei Ke, Aaron S. Brewster, Stella X. Yu, Daniela Ushizima, Chao Yang, and Nicholas K. Sauter. A convolutional neural network-based screening tool for x-ray serial crystallography. *Journal of Synchrotron Radiation*, 25(3):655–670, 2018.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Toronto, 1993.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *International Conference on Neural Information Processing Systems (NIPS)*, 2012.

Meghana Dinesh Kumar, Morteza Babaie, Shujin Zhu, Shivam Kalra, and Hamid R Tizhoosh. A comparative study of cnn, bovw and lbp for classification of histopathological images. In *Symposium Series on Computational Intelligence (SSCI)*, pp. 1–7, 2017.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: Bandit-based configuration evaluation for hyperparameter optimization. In *International Conference on Learning Representations (ICLR'17)*, 2017.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision (ECCV)*, 2014.

David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.

Woon Bae Park, Jiyong Chung, Jaeyoung Jung, Keemin Sohn, Satendra Pal Singh, Myoungho Pyo, Namsoo Shin, and K-S Sohn. Classification of crystal structure using a convolutional neural network. *IUCrJ*, 4(4):486–494, 2017.

PyTorch. ResNet Implementation in PyTorch. `https://github.com/pytorch/vision/blob/master/torchvision/models/resnet.py`, 2018.

Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. In *International Conference on Neural Information Processing Systems (NIPS)*, 2015.

Francesco Stellato, Dominik Oberthür, Mengning Liang, Richard Bean, Cornelius Gati, Oleksandr Yefanov, Anton Barty, Anja Burkhardt, Pontus Fischer, Lorenzo Galli, Richard A. Kirian, Jan Meyer, Saravanan Panneerselvam, Chun Hong Yoon, Fedor Chervinskii, Emily Speller, Thomas A. White, Christian Betzel, Alke Meentsc, and Henry N. Chapmana. Room-temperature macromolecular serial crystallography using synchrotron radiation. *IUCrJ*, 1(4):204–212, 2014.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016.

authors omitted. DeepFreak Implementation in PyTorch. *link omitted for blind review*, 2018.

Vladimir N. Vapnik. *The nature of statistical learning theory*. Springer-Verlag, 1995.

Michael M Woolfson. *An Introduction to X-ray Crystallography*. Cambridge University Press, 1997.

Jun Yang, Yu-Gang Jiang, Alexander G. Hauptmann, and Chong-Wah Ngo. Evaluating bag-of-visual-words representations in scene classification. In *International Workshop on Multimedia Information Retrieval (MIR)*, 2007.

Margot Lisa-Jing Yann and Yichuan Tang. Learning deep convolutional neural networks for x-ray protein crystallization image analysis. In *AAAI Conference on Artificial Intelligence*, 2016.

Angelo Ziletti, Devinder Kumar, Matthias Scheffler, and Luca M Ghiringhelli. Insightful classification of crystal structures using deep learning. *Nature Communications*, 9(1):2775, 2018.

Table 7: Search space for the feature extractors and the RF/SVM classifiers hyperparameter search.

	Hyperparameter	**Type**	**Values**	**Default**
SVM	C	Ordinal	1 or 2^x for x in {-5, -3, ..., 13, 15}	1
	γ	Ordinal	0 or 2^x for x in {-15, -13, ..., 1, 3}	0
	Class weights	Categorical	None	None
			Balanced	
			[0.35, 0.35, 0.1, 0.1, 0.1]	
			[0.3, 0.3, 0.133, 0.133, 0.133]	
			[0.25, 0.25, 0.166, 0.166, 0.166]	
RF	Number of trees	Ordinal	{10, 100, 1000}	10
	Max features	Ordinal	$\{\sqrt{features}, 0.25, 0.5, 0.75\}$	$\sqrt{features}$
	Max depth	Ordinal	{None, 2, 4, 6, 8, 10, 20}	None
	Class weights	Categorical	None	None
			Balanced	
			[0.35, 0.35, 0.1, 0.1, 0.1]	
			[0.3, 0.3, 0.133, 0.133, 0.133]	
			[0.25, 0.25, 0.166, 0.166, 0.166]	
GLCM	Distances	Categorical	Any combination of {1, 2, 4, 5, 8}	5
	Angles	Categorical	Any combination of {0, 45, 90, 135}	0
LBP	Points	Ordinal	{4, 8, 16, 24}	24
	Radius	Ordinal	{0, 1, 2, 3}	3
SIFT	Clusters	Ordinal	{10, 25, 50, 100, 250, 500, 1000, 5000}	100

A SEARCH SPACE SVM AND RF WITH FEATURE EXTRACTORS

SVM and RF search spaces with the feature extractors are summarized in Table 7. The search space of the feature extractors includes the hyperparameters mentioned in Section 4.1. SVM search space includes the cost of misclassification parameter (C) and the γ parameter for the RBF kernel. RF search space comprises the number of trees in the forest, the maximum number of features used by the trees to find the best split, and the maximum depth of trees. The search spaces for both classifiers also include a "class weight" hyperparameter that assigns different weights to the entries classes. In the class weights, *None* indicates all classes have the same weight, *Balanced* shows all classes are weighted according to their number of samples, and the value arrays mean the weight given to each class of DiffraNet (from *blank* to *strong*).

B DEEPFREAK SEARCH SPACE

Our search space for DeepFreak includes some topologies and learning hyperparameters (Tables 8 and 9). The topologies we have designed for DeepFreak seek to increase the network downsampling, reducing training times and improving accuracy. Likewise, we search for the mix of initial learning rate, momentum, weight decay, and batch size that maximizes accuracy.

C DEEPFREAK BEST CONFIGURATIONS

Table 10 shows the three best topologies found by BOHB for DeepFreak.

D SVM AND RF CONFUSION MATRICES

Tables 11 and 12 show the confusion matrices for SVM and RF respectively.

Table 8: Possible topology adaptations for ResNet.

Name of Variant	**Default**	**1**	**2**	**3**	**4**	**5**	**6**
Number of Filters	64	64	64	64	64	64	64
1st convolution size	7	7	7	7	7	7	7
1st convolution stride	2	2	2	2	4	2	2
1st pool size	3	3	3	3	3	3	3
1st pool stride	2	2	2	2	2	2	2
2nd pool size	7	9	13	7	7	7	15
2nd pool stride	1	2	2	1	1	2	2
block strides	1, 2, 2, 2	1, 2, 2, 2	1, 2, 2, 2	2, 2, 2, 2	1, 2, 2, 2	2, 2, 2, 2	1, 2, 2, 2
Name of Variant	**7**	**8**	**9**	**10**	**11**	**12**	**13**
Number of Filters	64	64	8	8	64	16	16
1st convolution size	7	7	7	7	7	7	7
1st convolution stride	2	2	2	2	2	2	2
1st pool size	3	3	3	3	3	3	3
1st pool stride	2	2	2	2	2	2	2
2nd pool size	8	7	7	7	9	9	7
2nd pool stride	1	1	1	2	2	2	2
block strides	2, 2, 2, 2	1, 2, 3, 3	1, 2, 2, 2	1, 2, 2, 2	1, 2, 2, 3	1, 2, 2, 2	1, 2, 2, 2

Table 9: Search space for DeepFreak hyperparameter search

Hyperparameter	**Type**	**Values**	**Default**
Topology	Categorical	[1, 13]	3
Learning rate	Log Real	[1e-4, 10]	0.1
Momentum	Real	[0.5, 1]	0.9
Weight decay	Real	[0.00001, 0.00005]	0.00001
Batch size	Categorical	{4, 8, 16, 32, 64}	8

Table 10: Three best configurations found by BOHB.

	Topology	Learning rate	Momentum	Weight Decay	Batch Size
A	6	8.6673e-04	0.7770	2.5380e-05	1
B	8	8.4474e-04	0.56168	3.4855e-05	1
C	1	5.9409e-03	0.70739	3.0193e-05	2

Table 11: Confusion matrix of SVM for the test set.

		Predicted class					
		blank	noxtal	weak	good	strong	Recall (%)
True class	blank	2069	0	0	0	0	100
	noxtal	0	3266	1	0	1	99.93
	weak	1	160	3142	44	0	93.90
	good	0	0	40	2377	27	97.26
	strong	0	0	0	22	1450	98.51
	Precision (%)	100	95.33	98.71	97.3	98.11	

Table 12: Confusion matrix of RF for the test set.

		Predicted class					
		blank	noxtal	weak	good	strong	Recall (%)
True class	blank	2069	0	0	0	0	100
	noxtal	0	3266	2	0	0	99.94
	weak	0	53	3254	39	0	97.25
	good	0	0	45	2368	31	96.89
	strong	0	0	0	25	1447	98.30
	Precision (%)	100	98.4	98.58	97.37	97.9	