X-ray Scattering Image Classification Using Deep Learning

Boyu Wang[1], Kevin Yager[2], Dantong Yu[2], and Minh Hoai[1]

[1]Stony Brook University, Stony Brook, NY, USA

{boywang, minhhoai}@cs.stonybrook.edu

[2]Brookhaven National Laboratory, Upton, NY, USA

{kyager,dtyu}@bnl.gov

Abstract

Visual inspection of x-ray scattering images is a powerful technique for probing the physical structure of materials at the molecular scale. In this paper, we explore the use of deep learning to develop methods for automatically analyzing x-ray scattering images. In particular, we apply Convolutional Neural Networks and Convolutional Autoencoders for x-ray scattering image classification. To acquire enough training data for deep learning, we use simulation software to generate synthetic x-ray scattering images. Experiments show that deep learning methods outperform previously published methods by 10% on synthetic and real datasets.

1. Introduction

X-ray scattering is used in a wide variety of domains, from determining protein structure to observing realtime structural changes in materials. Broadly speaking, x-ray scattering can probe the physical structure of materials at the molecular and nanoscale. The technique consists of shining a bright, collimated x-ray beam through a material of interest; detailed information about structural order is then inferred from the far-field pattern of scattered rays [3]. The scattering images contain visual features, such as rings, spots, and halos, which encode detailed information about the size, orientation, and packing of atoms, molecules, and nanoscale domains [16]. Modern x-ray detectors can generate 50,000 to 1,000,000 images/day (1-4 TB/day); thus it is crucial to automate the image processing workflow as much as possible. The lack of immediate feedback during x-ray scattering experiments currently limits the scientific productivity of this technique. Manually curated image analysis becomes a bottleneck, due to the enormous diversity of possible image features; we propose instead to develop computer vision algorithms to automate the process of image analysis.

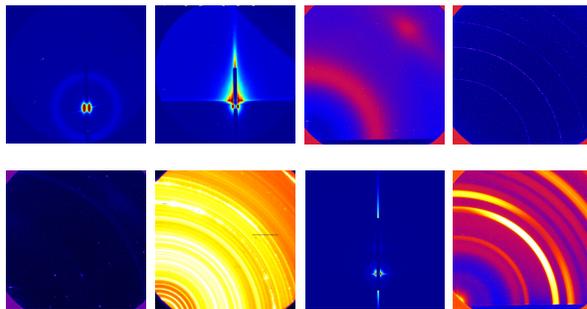Our task is to classify x-ray scattering image attributes.

Figure 1: Examples of x-ray scattering images. Images are shown using (arbitrary) false-colors; source images are grayscale. We explore the use of deep learning methods to automatically recognize the attributes of x-ray images. The first row of this figure shows some attributes that we want to recognize; from left to right, the attributes are: TSAXS, Linear Beamstop, Halo, and Powder. Automatic attribute recognition is a challenging problem due to the high intraclass variance. Images with the same attribute can look very different; the second row of this figure shows example images with the attribute 'ring.'

These attributes represent a diverse set of characteristics ranging from the type of measurement, e.g. 'small-angle x-ray scattering (SAXS)' or 'wide-angle x-ray scattering (WAXS)', to instrumental information, e.g. 'linear beamstop' or 'beam off image', to appearance-based scattering features, e.g., 'halo' or 'ring', to chemical composition and physical properties of the materials, e.g., 'powder' or 'SiO_2'. Figure 1 shows example images with the 'ring' attribute, which signifies the appearance of a circle or arc of high intensity. As these examples show, images with the same attribute can otherwise be strikingly different. This makes classification of x-ray images extremely challenging.

To the best of our knowledge, there exists little work for automatically analyzing x-ray scattering images. The most closely-related prior work is that of Kiapour et al. [6], which also aimed to recognize the set of image attributes consid-

ered in this paper. Their work used hand-designed features such as HOG [1] and SIFT [13]. Unfortunately, these types of features were designed for natural images instead of x-ray images. As a consequence, the method developed by Kiapour et al. [6] lacks the performance that would be desired for trustworthy automated analysis of real scientific data.

In this paper, inspired by the recent success of deep learning methods [9], we propose to investigate the use of deep learning for x-ray image classification. In particular, we propose to use Convolutional Neural Networks (CNN) [7, 10, 11] (in particular Residual Networks) and Convolutional Autoencoders [5] to extract features that are important for x-ray image classification. Convolutional filters are able to extract local patterns across whole images. With stacking of multiple convolutional layers, CNN is able to extract hierarchical features from images.

However, the size of the previously-available x-ray image dataset is quite small; far too small for robust application of deep learning methods. The dataset collected by [6] only contains 2832 images. Manually labeling x-ray images requires a significant amount of domain experts' time, which could otherwise be spent on high-level scientific discovery. It becomes infeasible for experts to tag a large dataset containing millions of images. In this paper, we propose a synthetic dataset which is generated based on the known physics underlying x-ray scattering experiments and ad hoc features to emulate the artifacts and defects present in experimental images. By utilizing the synthetic dataset, we can train deep networks for x-ray scattering images. The trained networks can be used to extract feature representations for both synthetic and real x-ray scattering images. Using this type of feature representation, we obtained a method that outperforms hand-crafted features [6] by 10% on both synthetic and real datasets.

The rest of the paper is organized as follows: In section 2 we introduce the real and synthetic datasets used in our experiments. In section 3, we describe the deep learning techniques for x-ray scattering image attribute classification. In section 4, we report the performance of our methods.

2. Datasets

We used two datasets for developing and evaluating deep learning methods for analyzing x-ray scattering images.

2.1. X-ray Materials Discovery Dataset (XMD)

X-ray Materials Discovery Dataset (XMD) [6] contains 2832 x-ray scattering images collected from thirteen x-ray scattering measurement runs. The number of images in each experiment varies between 54 and 618. The dataset includes a wide range of samples: nano-particles in solution, lithographic gratings, self-assembling polymer films, organic semiconductors, etc. All images are single-channel

with intensities in the range $[0, 2^{16}]$. The images have been tagged with 98 attributes by a domain expert. Images are labeled with an average of 11.7 attributes.

Since the images within the same measurement run can be much more similar than across different runs, cross validation should be performed by leaving one measurement run out [6]. In other words, we perform 13-fold validation; in each fold, we train the model on all runs except one and use the resulting model to test and predict on the one that is excluded from training.

Due to the unbalanced distribution of attributes—for example, SAXS and WAXS attributes may be prevalent in many samples however specific material attributes like SiO_2 is quite rare—classification accuracy is not a useful evaluation metric. Following [6], we use Average Precision (AP) for evaluation, which is more appropriate for unbalanced data than accuracy.

2.2. Synthetic Dataset

Deep learning methods generally require large and diverse training sets to yield good performance. Unfortunately, the available human-tagged experimental datasets are very small. To effectively exploit deep learning methods for x-ray scattering image classification, we propose using large datasets with synthetic scattering images. We implemented our own simulation software to generate synthetic datasets by mixing ad-hoc methods and physics-based simulations of scattering data. In the case of ad-hoc images, relevant features such as rings or spots are generated and summed together. In the case of simulations, we combine a variety of well-known methods, including using known analytic models for the expected scattering of certain objects (e.g., spherical nanoparticles) or assemblies (e.g., cubic lattice of objects). The simulation code also includes a module allowing the scattering to be computed for arbitrary arrangements of entities that represent atoms, molecules, or nanoparticles. A particular synthetic image is generated by selecting a random set of simulation modules, and summing together their outputs (based on randomly-selected input variables). The final image is then adjusted to simulate a variety of experimentally-realistic effects, including background noise, shot noise, gaps and shadows arising from experimental geometry. Overall, the simulated images cover a wide space of possibilities, both in terms of the types of observed structures/patterns and the quality of images.

The simulation code allows the generation of an arbitrary number of training images; where each image carries the appropriate tags (which are known based on the selected simulation modules). In the present experiments, we generate a synthetic dataset which contains 100,000 x-ray scattering images. Figure 2 visualizes the differences between synthetic images and experimental images. We observe that synthetic images and real experimental images are visually
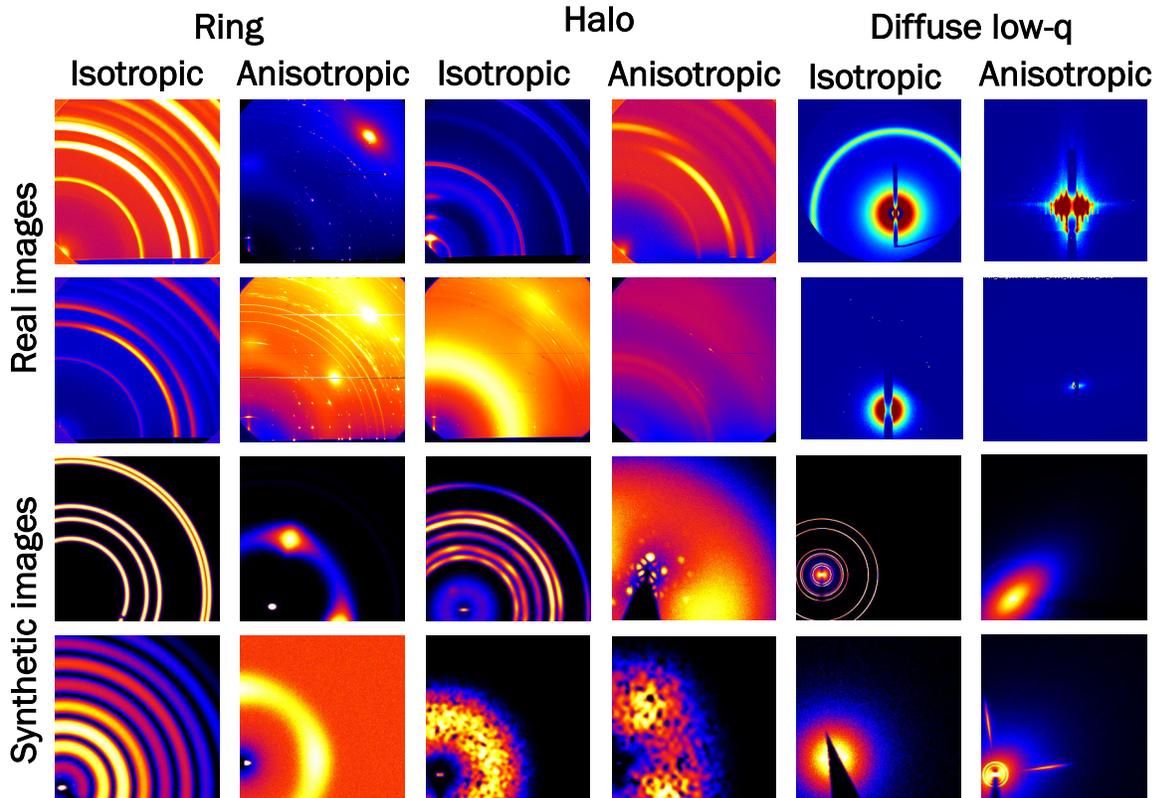
Figure 2: Comparison between synthetic images and real experimental images. The first and second rows are real experimental images, while the third and forth rows are synthetic images. Images in the same column have the same attribute. From left to right, the attributes are: Ring: Isotropic, Ring: Anisotropic, Halo: Isotropic, Halo: Anisotropic, Diffuse low q: Isotropic, and Diffuse low q: Anisotropic. Visually, synthetic and real images are indiscernible.

similar. This suggests the potential usefulness of synthetic data as training examples.

3. Deep-learning Methods

In this section, we describe two deep learning techniques to extract features for x-ray scattering images. Rather than using hand-crafted features, which are often designed for natural images, we use Convolutional Neural Networks to automatically extract features that are important for x-ray images. Once the features are extracted, we train one vs. all support vector machines (SVMs) classifiers [15] to predict x-ray image attributes. We adopt two methods for feature extraction: one is based on supervised learning, and the other is unsupervised learning. In rest of this section, we will discuss the two methods in details.

3.1. Residual network

We train a Convolutional Neural Network (CNN) on the synthetic dataset to classify x-ray image attributes. We use the recently proposed 50-layer Residual Network [4] as our
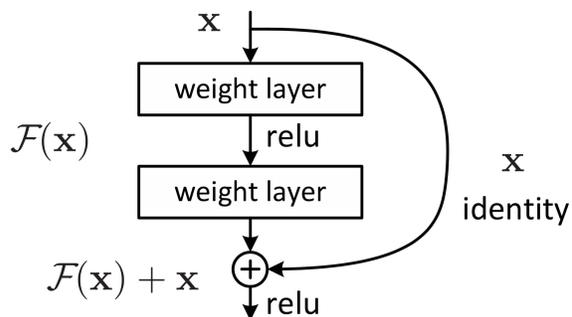

Figure 3: The building block of residual networks [4]. With network bypasses, residual networks explicitly reformulate the layers as learning residual functions with reference to the layer inputs.

architecture. Figure 3 shows the basic learning block in the residual network. Using bypass connections, a residual network explicitly reformulates the layers as learning residual

layer name	output size	kernels	
conv1	112×112	7×7, 64, stride 2	
conv2_x	56×56	3×3 max pool, stride 2	
		$1 \times 1, 64$ $3 \times 3, 64$ $1 \times 1, 256$	× 3
conv3_x	28×28	$1 \times 1, 128$ $3 \times 3, 128$ $1 \times 1, 512$	× 4
conv4_x	14×14	$1 \times 1, 256$ $3 \times 3, 256$ $1 \times 1, 1024$	× 6
conv5_x	7×7	$1 \times 1, 512$ $3 \times 3, 512$ $1 \times 1, 2048$	× 3
pooling	1×1	average pooling	
fc	1×1	2048×num of attributes	

Table 1: The architecture and parameter settings of the convolutional neural network developed here. It is based on the 50-layer residual network [4].

functions with regard to the input layer. By adopting such a framework, deeper networks are easier to optimize than those without bypass connections, gaining accuracy from a considerably deep architecture. Table 1 shows the detailed architecture of our adopted network. We modify the softmax layer to a binary sigmoid layer because image attributes are not mutually exclusive. The dimension of network output is equal to the number of attributes, and each element in the output vector represents the probability of having that attribute. The final loss function is the summation of the losses incurred by each attribute.

We train the residual network on a synthetic dataset with 100,000 x-ray images. During training, we did not train the network to predict the entire set of attributes to avoid the problem of unbalanced data: some attributes appear in the majority of images while some attributes are really rare. We pick a subset of 17 attributes which are not rare and guarantee that every x-ray image has at least one of these 17 attributes. Table 2 lists the 17 selected attributes. Once the network has been trained, we can use it to extract the feature representation for an image patch as follows. First, the image patch is resized to 224×224, and subsequently fed into the network, and secondly, the activation of the network right before the output layer is taken as the feature vector representation. This feature vector has 2048 dimensions, and is conventionally referred to as fc7 feature vector [7].

Figure 4 visualizes the first layer's filters learned by our residual network. As can be seen, many filters of the the first layer tend to pick up edge-like signals.
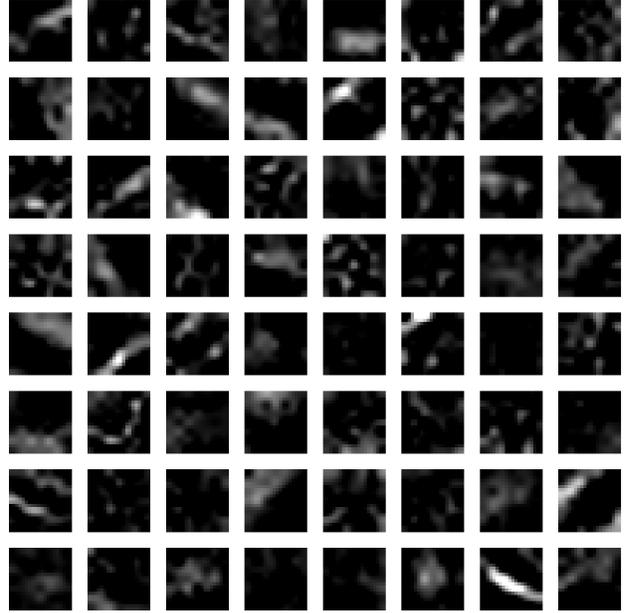

Figure 4: Visualization of the filters of the first layer of the learned residual network.

BCC	Beam Off Image	Circ. Beamstop
Diffuse high-q	Diffuse low-q	FCC
Halo	High background	Higher orders
Linear beamstop	Many rings	Polycrystalline
Ring	Strong scattering	Structure factor
Weak scattering	Wedge beamstop	

Table 2: The list of 17 attributes for training the residual network. These attributes were selected to be representative of the diversity of labels associated with x-ray scattering images.

We use the learned network to extract feature vectors for the real experimental XMD dataset. The size of each images in this dataset is 1024×1024, which is larger than the expected input size of the the CNN, which is 224×224. To compute the feature vector representation, we first resize a 1024×1024 image to three different scales: 256×256, 384×384, and 512×512. At each scale, we crop five images of size 224×224 (at the center and four corners), and feed them into the previously-trained network to obtain the corresponding feature vector. We average the corresponding fifteen feature vectors as the final feature representation for the x-ray image.

3.2. Convolutional Autoencoder network

The second method to extract feature vectors is the convolutional autoencoder. Instead of performing autoencoding on full-size images, we perform autoencoding on image
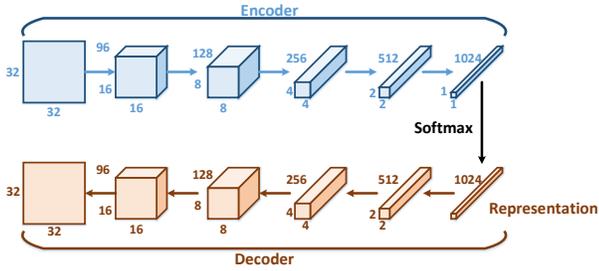
Figure 5: Architecture of the convolutional autoencoder. The difference between this architecture and the traditional autoencoder is that we add a softmax layer after the output of encoder and we use this vector, of which the size is 1024, as the feature vector for an image patch.
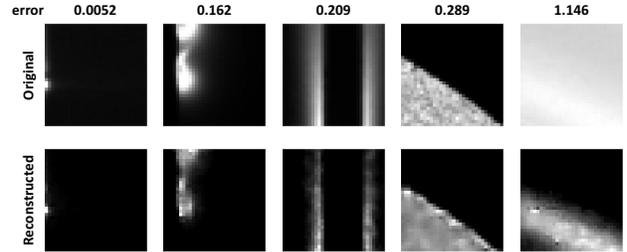

Figure 6: Visualization of original image patches (top row) and reconstructed image patches (bottom row) using convolutional autoencoder. From left to right, the reconstruction errors are 0.0052, 0.162, 0.209, 0.289 and 1.146. Here the reconstructed images retain the important visual structures of the original images even when the reconstruction error is high (at the right most column).

patches because: i) the size of original images is too big to train an autoencoder; and ii) downsampling original image to a smaller resolution such as 256×256 or 128×128 may lose some important details, such as sharp, localized peaks. Since we train the autoencoder at the patch level, even though we have a limited number of real experimental images (2832 images in XMD dataset), we still obtain many image sample patches from each image. Therefore, our convolutional autoencoder network is trained on real experimental image patches. We resize 1024×1024 images to multiple scales and randomly extract 1000 32×32 patches per image as the training set for autoencoder. Figure 5 shows the architecture of our autoencoder for image patches. The difference between our autoencoder and traditional autoencoder is that we applied a softmax layer after the output of encoder and used the output vector with the size of 1024 as the feature vector for input image patch. The objective here is to use the autoencoder to cluster image patches. The dimension of the encoder output represents the number of clusters, and each element belongs to a data cluster. After applying the softmax function, the vector represents the possibilities that the input image falls into different data clusters.

In our experiment, the learned autoencoder has the minimum reconstruction error of 0.0044, the maximum of 1.2510, and the average of 0.6817. Figure 6 shows the original in the top row and reconstructed image patches at the bottom row. For left to right, the reconstruction error (the difference between the top image and the bottom image) increases. The reconstruction retains the important visual structures of the original images.

Representation vectors have 1024 dimensions. By setting the representation vector to be one-hot vector (all values are 0 expect one location is 1), we get the data cluster represented in that location. By changing the location, we can get 1024 different data clusters. Figure 7 visualizes 20 data clusters. These data clusters capture the edges with

different angles and the blobs at different locations.

After training the autoencoder, we perform a spatial pyramid matching (SPM) [8] with three levels to extract the feature vector descriptor for an image. The SPM partitions an image into sub-regions of increasingly fine granularity and computes the histograms of local features found within each sub-region. This approach allows the classifier to better understand the spatial relationship between different areas of an image. We use sum pooling because we compute histograms and intent to obtain the frequency of each cluster.

4. Experimental results

In this section, we report the performance of the deep-learning features for x-ray image classification. We report the average precision values on both synthetic and real datasets, and compare them with previously published results.

4.1. Performance on synthetic dataset

We first evaluate the performance of the residual network on the holdout (test subset) of the synthetic dataset (note that we split the synthetic dataset into two disjoint train/test subsets). For the test data, the residual network achieves the mean average precision of 77.1% for the 17 attributes given in Table 3 (note that we trained the residual network on these 17 attributes only). The result is given in Table 3. We compare this result with a shallow-learning method that is based on the Bag-of-Word approach [12, 14]. This approach consists of: i) using k-means to learn a visual codebook; ii) applying spatial pyramid pooling to generate a feature vector representation for each image; iii) training a binary SVM for each of 17 attributes in consideration. This approach only achieves a mean Average Precision of 67.1%, which is 10% lower than our method using the feature de-
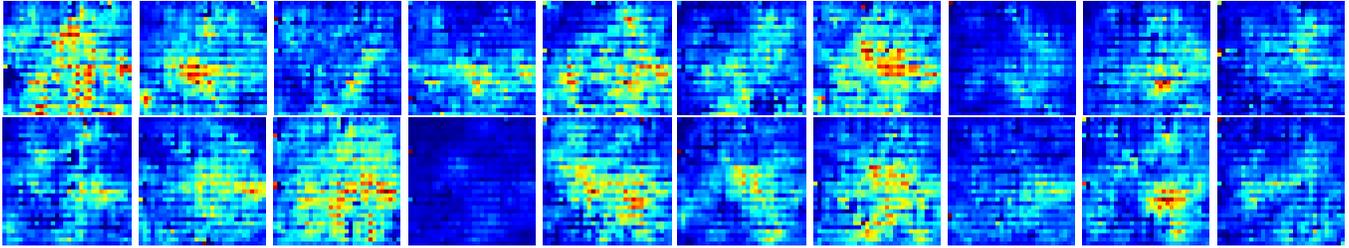
Figure 7: Visualization of clusters. These data clusters capture edges of different angle, and blobs of different location.

Method	mAP
k-means + Bag-of-Words + Spatial Pyramid	67.1
Residual network (deep learning)	77.1

Table 3: Mean Average Precision (mAP) on synthetic dataset for 17 attributes

(a) Ring   (b) Peaks

Figure 8: Precision-recall curves for recognizing attributes 'Ring' and 'Peaks'.

Feature	mAP
lbpphog [6]	51.5
Residual network	59.5
Patch autoencoder	57.9
Residual network + Patch autoencoder	61.1

Table 4: Mean Average Precision (mAP) for all 98 attribute classifiers on the XMD dataset. Deep-learning features (Residual network and Path autoencoder) outperform hand-designed features by a large margin. The best result is achieved when two types of deep-learning features are combined.

scriptors from deep-learning. This indicates the benefits of using deep-learning for x-ray image classification.

4.2. Performance on XMD dataset

For XMD dataset, we use both the residual network and the convolutional autoencoder to extract features for each image, then train a multi-class SVM [2] with a one vs. all approach on all 98 attributes. Following [6], we report the performance on these 98 attributes in Table 4. The residual network features achieve 59.5% mean AP, which outperforms hand-crafted features by 8%. The convolutional autoencoder features achieve a mean AP of 57%. Combining the features generated both the residual network and convolutional autoencoder, we obtain the mean AP of 61.1%, outperforming hand-crafted features by 10%. In [6] they also report the performance using a hierarchical classifier in which the first classifier is used to separate images into two categories: small-angle (SAXS) and wide-angle (WAXS), and subsequently the second classifier is used to classify the fine-grained attributes in each category. The hierarchical approach boosts the performance from 51.5% to 55.5%. In our method, we have not used the hierarchical approach, and we hypothesize that the hierarchical approach will also improve the performance of our method. One of our future works is to build a more fine grained hierarchical structure based on attributes correlations to boost the performance.

Analyzing the XMD data, we discovered that 18 out of 98 attributes only appear in a single experiment run (of the 13 experiment runs). Therefore, for the leave-one-experiment-out cross validation procedure, these attributes will only appear in either the training set or test set. When an attribute does not appear in the test set, the average precision for this attribute is one. When the attribute does not
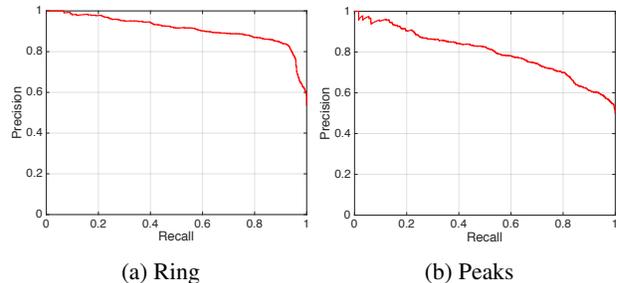
appear in the train set, the average precision is equal to the average precision of a random classifier. In both cases, the results do not depend on the type of the feature representation. Herein, we suggest to remove these attributes from the experimental analysis to obtain a more indicative mean average precision value. The mean average precision on the 80 remaining attributes is shown in Table 5. Figure 8 shows precision-recall curves of the method that uses residual network features for recognizing 'rings' and 'peaks'.

We also compared the performance of using hand-crafted features and deep learning features for 30 attributes of which the APs are explicitly reported in [6]. We consider the AP gap, which is the difference between the AP obtained by using deep learning features and the AP obtained by using the hand-crafted Ibpphog [6]. Figure 9 shows

Feature	mAP
Residual network	59.03
Patch autoencoder	56.00
Residual network + Patch autoencoder	59.95

Table 5: Mean Average Precision (mAP) for 80 attributes. Each attribute in this experiment appears in at least two experiment runs of the XMD dataset.
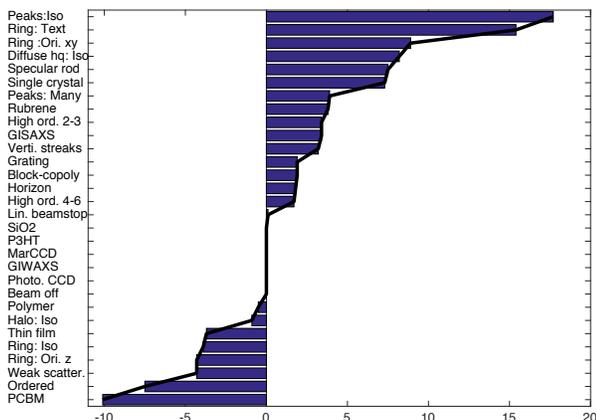

Figure 9: **Average precision (AP) gap.** This shows the difference between the average precision of deep learning features and hand-crafted features. The AP of deep learning features are higher than or equal to the AP of hand-crafted features for the majority of attributes.

AP gap for all 30 attributes. From the figure, we observe that the deep learning features outperform the hand-crafted features by a large margin for detecting multiple attributes, including Specular rod, Peaks: Isotropic, Ring: Textured, High order 2-3, Ring: oriented xy, Vertical streaks, Single crystal, Block-cropoly, Peaks: Many, Grating, Diffuse high-q: Isotropical, High order 4-6, and Rubrene. However, for the attributes such as Ordered, Ring: oriented z, Ring: Isotropic, PCBM, and Weak Scattering, the deep learning features do not perform as well as the hand-crafted features. We note that both approaches (deep learning and hand-crafted) perform well for image attributes where the possible options are disjoint and highly distinct. For instance, detecting the type of detector used in the experiment (MarCCD vs. Photonics CCD) is an 'easy' task for both approaches. The hand-crafted approach appears to slightly out-perform deep learning on a small number of attributes, especially those that denote a rather vague interpretation of the overall image (e.g., ordered). Conversely, deep learning achieves a much-improved performance for attributes denoting distinct localized features such as Specular rod and Peaks: Isotropic. Importantly, deep learning can evidently identify the attributes associated with combining a number of disparate features throughout image such as single crystal; that is, we confirm that the hierarchical, multi-level internal representations of deep learning are well aligned with the complex, multi-feature labels that are frequently used by domain experts to describe x-ray scattering images.

Figure 10 shows some images where our method using deep-learning features do not agree with human annotation. The first two rows show some false positives (which means a human annotator tags this image as negative for this attribute, but our method predict it with a high scores for this attribute). The third and fourth rows show some false negative results (which means a human annotator tags this image as positive for this attribute, but our method labels it with a low score for this attribute). After consulting with domain experts, images in the first row reflect some human annotation error in tagging original images, i.e., our method successfully identified human annotation errors. Some images are ambiguous even for a human to detect, e.g., diffuse high-q scattering is weak and spatially distributed. Even human experts may disagree with respect to the marginal (weakly-scattering) examples of this feature. Nevertheless, our approach failed to classify some images because the target attribute has unusual appearance or is highly localized: for example, the positive image with the 'Ring' attribute the the third row and second column. It is rare to see a full ring appearing in this type of experiment (GISAXS). The lack of training examples potentially explains why this atypical scattering pattern was mis-classified. The image on the third row and first column is another mis-classified example that is rare and contains a positive 'Thin film' attribute. This is indeed a measurement of a thin film. However, the common visual features, such as a distinct horizontal stripe, are not present possibly because of the misalignment during the experiment. Only the subtle hints exist to help a trained human expert correctly classify it as a thin film measurement. There is a broad class of images that even a human expert finds extremely challenging to classify, owing to the subtlety of the features, the ambiguity of the tag, or the violation of standard experimental assumptions (e.g., an error occurs during experiments). Consequently these types of images are challenging to any deep learning based classification approach. By augmenting training sets with some examples of borderline cases, we can improve the classification accuracy for these atypical images. Thus, one potential future work is to augment our simulation code to generate synthetic images exhibiting a variety of atypical or marginal patterns.

5. Conclusions

We have explored the use of deep learning methods for automatic recognition of x-ray scattering images' attributes. To overcome the size limitation of available annotated x-ray image datasets, we used simulation software to gener-

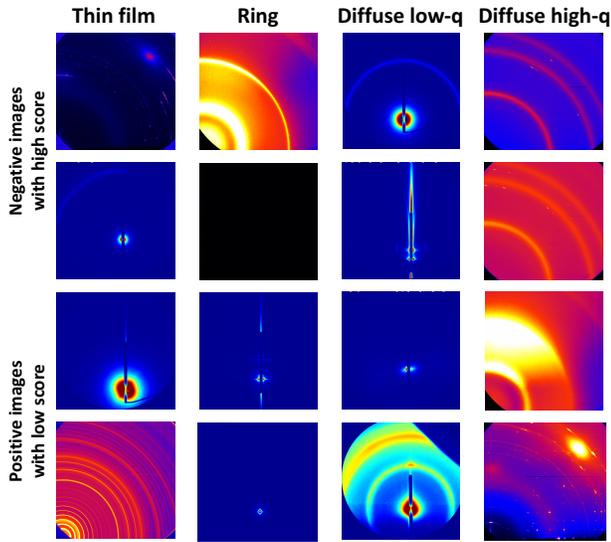| | Thin film | Ring | Diffuse low-q | Diffuse high-q |

Figure 10: Example images of mismatch between the prediction of our methods, and the human-assigned tag. The first two rows show negative images with high score (i.e., a human annotator tags this image as negative for this attribute, but our prediction method predicts high scores for this attributes). The third and fourth rows show positive images with low score (i.e., a human annotator tags this image as positive for this attribute, but our prediction method predicts low scores for this attributes). Attributes from left to right are: Thin film, Ring, Diffuse low-q, Diffuse high-q.

ate synthetic x-ray images for training. Our features are based on fully connected layer output of a residual network and the representation layer of a convolution autoencoder. Evaluations on both synthetic and real datasets show that deep-learning features outperform hand-crafted features by a large margin of 10%, using mean average precision as the evaluation metric.

References

[1] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proc. CVPR*, 2005.

[2] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research*, 9:1871–1874, 2008.

[3] A. Guinier. *X-ray diffraction in crystals, imperfect crystals, and amorphous bodies*. Courier Corporation, 1994.

[4] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[5] G. Hinton and R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[6] M. H. Kiapour, K. Yager, A. C. Berg, and T. L. Berg. Materials discovery: Fine-grained classification of x-ray scattering images. In *IEEE Winter Conference on Applications of Computer Vision*, pages 933–940. IEEE, 2014.

[7] A. Krizhevsky, I. Sutskever, and G. Hinton. ImageNet classification with deep convolutional neural networks. In *NIPS*, 2012.

[8] S. Lazebnik, C. Schmid, and J. Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. CVPR*, 2006.

[9] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[10] Y. LeCun, B. Boser, J. S. Denker, and D. Henderson. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989.

[11] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[12] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, 2001.

[13] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004.

[14] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003.

[15] V. Vapnik. *Statistical Learning Theory*. Wiley, New York, NY, 1998.

[16] K. G. Yager, Y. Zhang, F. Lu, and O. Gang. Periodic lattices of arbitrary nano-objects: modeling and applications for self-assembled systems. *Journal of Applied Crystallography*, 47(1):118–129, 2014.