Robust and Scalable Deep Learning for X-ray Synchrotron Image Analysis

Nicole Meister[1*], Ziqiao Guan[2*], Jinzhen Wang[3], Ronald Lashley[4], Jiliang Liu[5], Julien Lhermitte[5], Kevin Yager[5], Hong Qin[2], Bo Sun[6], Dantong Yu[3]

[1]Centennial High School, MD
nicolemeister99@gmail.com

[2]Stony Brook University, NY
{ziguan, qin}@cs.stonybrook.edu

[3]New Jersey Institute of Technology, NJ
{jw447, dtyu}@njit.edu

[4]Lincoln University, PA
ronald.lashley@lincoln.edu

[5]Brookhaven National Laboratory, NY
{Jiliangliu, lhermitte, kyager}@bnl.gov

[6]Rowan University, NJ
sunb@rowan.edu

Abstract*—X-ray scattering is a key technique in modern synchrotron facilities towards material analysis and discovery via structural characterization at the molecular scale and nano-scale. Image classification and tagging play a crucial role in recognizing patterns, inferring meaningful physical properties from sample, and guiding subsequent experiment steps. We designed deep-learning based image classification pipelines and gained significant improvements in terms of accuracy and speed. Constrained by available computing resources and optimization library, we need to make trade-off among computation efficiency, input image size and volume, and the flexibility and stability of processing images with different levels of qualities and artifacts. Consequently, our deep learning framework requires careful data preprocessing techniques to down-sample images and extract true image signals. However, X-ray scattering images contain different levels of noise, numerous gaps, rotations, and defects arising from detector limitations, sample (mis)alignment, and experimental configuration. Traditional methods of healing x-ray scattering images make strong assumptions about these artifacts and require hand-crafted procedures and experiment meta-data to de-noise, interpolate measured data to eliminate gaps, and rotate and translate images to align the center of samples with the center of images. These manual procedures are error-prone, experience-driven, and isolated from the intended image prediction, and consequently not scalable to the data rate of X-ray images from modern detectors. We aim to explore deep-learning based image classification techniques that are robust and capable of leverage high-definition experimental images with rich variations even in a production environment that is not defect-free, and ultimately automate labor-intensive data preprocessing tasks and integrate them seamlessly into our TensorFlow based experimental data analysis framework.

Keywords—X-ray Scattering, Image Analysis, TensorFlow, Deep Learning, Convolutional Networks

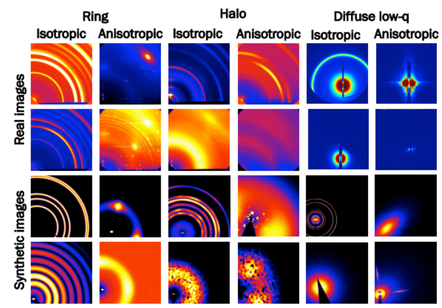

Fig. 1. Comparison of real experimental images (top 2 rows) and synthetic images (bottom 2 rows) [1].

I. INTRODUCTION

Modern synchrotron facilities are used worldwide to analyze the structures of materials at the molecular and nano-scale. Synchrotrons produce x-ray scattering images that are essential in characterizing the features and attributes of molecules. X-ray scattering images help scientists expose molecular structures at the atomic level and will ultimately enhance their understanding of the functions of material. With this knowledge, they can study how molecules interact with other molecules, how molecules undergo conformational changes [2], how enzymes perform catalysis [3], and so much more. This structural knowledge provides vital information in helping scientists design novel drugs that bind to protein to treat diseases [4].

To produce x-ray scattering images, the synchrotron first produces extremely bright synchrotron light (x-ray and infrared light) by accelerating electrons in a magnetically confined ring and then deflecting electrons with a strong magnetic field. Optical systems (silicon mirrors, apertures, and crystals) focus the synchrotron X-ray into a dense beam for probing samples. The x-ray or synchrotron light is directed at the sample chamber, holding the molecule to be analyzed. The x-ray is scattered off the molecules in the sample and constructively and destructively interfered with each other to form images with distinct visual features such as rings, spots, or halo. These visual patterns provide scientists insight into

* Equal contribution.

978-1-5386-3161-4/17/$31.00 ©2017 IEEE

the molecular structure, including the size, orientation, and packing of atoms (See Fig. 1).

National Synchrotron Light Source II (NSLS-II) at Brookhaven National Laboratory (BNL) is the world's brightest synchrotron light source with 60 available beamlines, enabling this advanced instrument to collect 50,000 to 1,000,000 images per day (1-4 terabytes per day) [5]. Due to this massive amount of data, scientists now face a daunting big data problem where the image streams significantly outpace any manual image analysis efforts on all the x-ray scattering images. On top of that, manual image analysis is extremely complex and consumes beam scientists' valuable time. This laborious process distracts scientists from thinking of the science in the experiment because they spend too much time attempting to analyzing a small fraction of the images out of the millions being produced.

This current data challenge limits the scientific productivity of the NSLS-II instrument because it is unable to perform at its full capacity. To keep up with the demanding data acquisition and to eventually move toward an automated experimentation process, scientists at BNL are taking advantage of deep learning methods, such as Convolutional Neural Networks (CNN) to process x-ray scattering images, to classify these images accurately and detect relevant features from them. As the amount of data increases, deep learning techniques outperform existing learning algorithms.

In 1989, LeCun published a paper introducing a biologically inspired neural network that uses convolutions, a linear operation, in place of general matrix multiplication [6]. Convolutional Neural Networks (CNNs) are specifically useful for processing data that has a "grid-like topology", such as image that consists of a 2-D grid of pixels. Recently, Wang et al. applied deep learning to x-ray scattering image recognition by implementing Convolutional Neural Networks (CNNs) and Convolutional Autoencoders [1]. They used simulation software to generate pre-tagged synthetic x-ray scattering images and then trained a CNN with 100,000 x-ray scattering images. Their experiments show that this deep learning method outperformed previously published methods by 10% on synthetic and real datasets.

Another issue in synchrotron instruments is noise in the images, deriving from detector limitations, sample (mis)alignment, experimental configuration, the x-ray beam hitting miscellaneous aerosol particles, or blurry optical lenses. Traditionally, data scientists utilize image healing methods. However, those methods of healing x-ray scattering images make strong assumptions about these artifacts and require hand-crafted procedures and experiment meta-data to de-noise, interpolate measured data to eliminate gaps, and rotate and translate images to align the center of samples with the center of images. These manual procedures are error-prone, experience-driven, and isolated from machine-based intended image prediction, and consequently not scalable to the data rates of X-ray images from modern detectors. Fig. 2 shows that the current Image CNN developed by Wang et al. is not robust to noise and the CNN network stack suffers the problem of deteriorating precision when the amount of the noise increases.

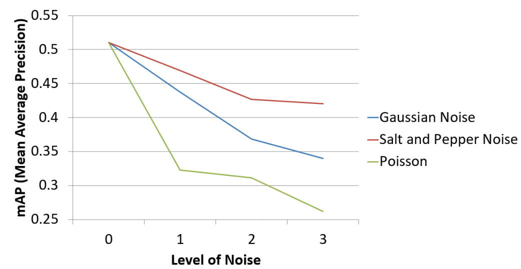

Fig. 2. Impact of noise on CNN accuracy.

Many machine learning specialists study regularization techniques of generalizing an algorithm to classify new inputs that have different patterns and distribution from the known training dataset. One form of regularization is data augmentation or noise injection. In previous works, researchers have found that injecting noise into the sample data can lead to neural networks with improved performance, meaning that the designed neural networks can have smaller misclassification errors than those without noise injection [7]. In this paper, we look into training the CNN with three different types of noise and analyzing the performance of the neural network in analyzing images with noise.

In addition, our system is based on Google TensorFlow, an open source software library for machine learning. Its flexible graph-representation architecture makes it easy to deploy computation to one or more CPUs or GPUs on desktops or servers, while the details of using CUDA and cuDNN are transparent to users. We are proposing a distributed computation paradigm that takes advantage of this feature.

II. Materials and Methods

A. CNNs

Convolutional Neural Networks are specifically useful for processing data that has a "grid-like topology" such as "image data, which can be thought of as a 2-D grid of pixels". In this experiment, we use the Google TensorFlow's deep CNN. This model consisted of layers with alternating operations of convolutions and nonlinear activations, followed by fully connected layers, and ending with a softmax classifier. To train the CNN, the images were passed through the initial convolutional layers and a sigmoid output layer. The prediction loss, or the distance between the predictions and true class, is computed by cross entropy. The loss function is defined as the sum of the cross entropy loss and the weight decay terms. The model uses stochastic gradient decent algorithm to minimize the loss of the function and improve its ability to classify images. During the process of training the model, the training terminated when the loss was below 0.10. To evaluate the CNN's ability to classify images, the network would calculate the precision, more specifically, how often the top prediction from the output layer matches the true label of the image.

B. Synthetic Dataset

A current challenge in deep learning for x-ray scattering images is the fact there is very limited number of humantagged experimental datasets. Although the images themselves are produced at an unprecedented rate, the amount of labelled data is still very small because specialists must manually go through images and tag them. Deep learning techniques benefit immensely from large data because there are more images to learn from. This shortage of labelled data prompts us to use synthetic data that can effectively mimic the patterns in the real images generated by x-ray interference beam. Fig. 1 shows that the synthetic images look very similar to the real data from NSLS-II.

C. Data Augmentation

Three forms of noise were applied to image: Gaussian Noise, Salt and Pepper Noise, and Poisson Counting Statistics. The forms of noise were selected because of their real world connection to the noise that a real image from NSLS-II may contain. Fig. 3 shows an example of the noisy images that are produced.

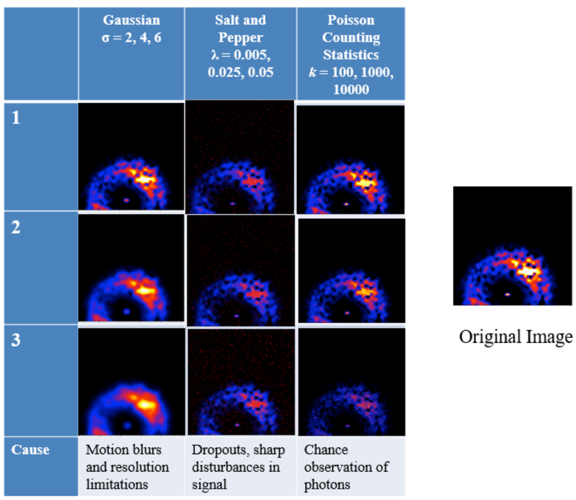

Fig. 3. Noise and image degradation techniques used for data augmentation.

1) Gaussian Noise

Gaussian noise is a standard form of noise that has been widely used to verify model robustness in signal, image and geometry processing. Zur et al. found that injecting neural networks with gaussian noise reduced the overfitting problem to a greater degree than the weight decay [8] and Goodfellow et. al described Gaussian noise as a common example of data augmentation to reduce overfitting [9]. We used the Gaussian noise in this process and render the CNN to be robust to this type of noise because of the following reasons: 1) the success in previous papers that use the Gaussian noise for data augmentation, and 2) the fact that the NSLS-II produces slightly blurry images and the Gaussian filter can closely represent this type of image distortion. To simulate this blurriness, the image pixels from the x-ray scattering image took on values that were Gaussian-distributed with varying σ's to change the amount of blurriness. The neural network was trained on images with many different values of sigma ranging from 0.5 to 6. For the evaluation process, the neural network was tested on 5,000 images that had three levels of gaussian noise, i.e., $\sigma = 2, 4, 6$.

2) Salt and Pepper Noise

The second form of noise applied to the image is Salt and Pepper noise in which the image contains sparsely occurring black and white pixels. This type of noise may occur when the x-ray beam hits an aerosol particle or mistakenly deflects off equipment. To model this form of noise, we randomly choose values in image array to corrupt by setting them to either 0 or 2^{16}, subject to a Poisson distribution with a probability mass function defined as follows:

$$f(k; \lambda) = \lambda^k e^{-\lambda} / k! \tag{1}$$

When training the network, the images had a random amount of salt and pepper noise (λ = 0.005-0.05). When evaluating the CNN performance, three levels of this noise were applied with the lambda of the distribution set to 0.005, 0.025, or 0.05 to see how well the network could classify images with different amounts of salt and pepper noise.

3) Poisson Counting Statistics

The third form of noise applied to the images is Poisson Counting Statistics. This form of noise originates from the particle nature of light where the probability of capturing the photon is a chance observation. When a synchrotron light source emit a stream of photos that scatter after directed towards a molecule, the probability of observing a photon hitting the detector is a chance observation. Typically, when there are millions of photons hitting a surface, the variation in number of photos is insignificant. However, synchrotron instruments have less photons and thus the variation in the number of photons is significant [10]. With x-ray beam that is scattered through the particle, there is a chance of that photon hitting the detector. With an image that has a high intensity pixel, there is a higher chance that the value of that pixel will be varied. This variation in the pixels can be modeled by resampling the pixel intensities with a Poisson distribution. At each individual pixel p, the scale factor λ is set to p/k where k is 100, 1,000, or 10,000. The division factor k controls the intensity of Poisson counting effect because a low mean λ reduces the observation probability and causes more severe loss of pixel value.

D. Training and Testing

To test the effects of data augmentation on the neural network's ability to classify x-ray scattering images, the network had to be first trained with a synthetic image dataset that contained with 50,000 noise-free x-ray scattering images.

Then, 5,000 x-ray scattering images were generated and corrupted with three levels of Gaussian noise, salt and pepper noise, and Poisson Counting Statistics. The CNN had never seen this testing set and the purpose was to use this set to evaluate the performance of the neural network. There were ten versions of 5,000 testing dataset: a noise-free version, three levels of Gaussian noise, three levels of salt and pepper noise, and three levels of Poisson counting statistics. These datasets were tested in the neural network trained with 50,000 noise-free x-ray scattering images to see how well the network classified the images before the network was trained with noisy images.

To train the CNN with augmented data, the CNN was trained on a total of 100,000 images. In addition to the previous 50,000 noise-free images, 50,000 new x-ray scattering images were generated and gaussian noise was applied to 1/3 of the images, salt and pepper noise was applied to 1/3 images, and Poisson counting statistics was applied to the remaining 1/3 images. After training the CNN on this new dataset, we evaluated its ability to classify the noise free images, three levels on each of three types of noise: Gaussian noise, salt and pepper noise, and Poisson Counting Statistics.

In addition, we also looked into training the CNN on images that had all three forms of noise applied to each image. In this case, we trained the CNN on the original noise-free 50,000 images and the 50,000 images that had all three forms of noise on each image. We then evaluated its performance when classifying the noise free images, three levels of Gaussian noise, salt and pepper noise, and Poisson Counting Statistics.

III. Multi-GPU Training

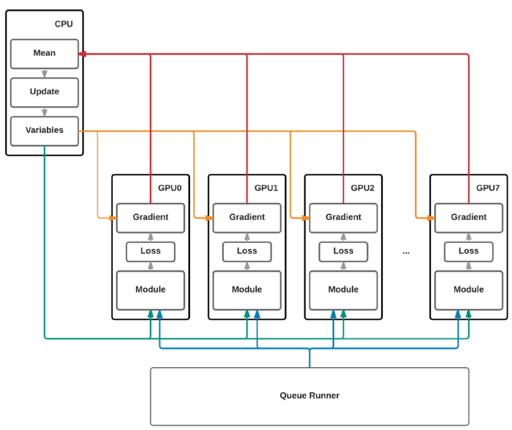

Fig. 4. Multi-GPU computation paradigm.

TensorFlow is an open source software library for machine learning. Its flexible graph-representation architecture makes it easy to deploy computation to one or more CPUs or GPUs on desktops or servers while all the implementation details in CUDA and cuDNN are transparent to users. Therefore, it is straightforward to migrate a Tensorflow application from CPU to GPU and from a standalone single processor pipeline to parallelel workflow with multiple GPU processors. We are proposing a distributed computation paradigm leveraging the parallelization and distributed computing capability of Google TensorFlow.

Fig. 4 shows our computation paradigm: it employs an in-graph data-parallelization approach in which the entire module is represented as a big graph and one copy of module, called a *replica*, along with a subset of training data, called a *batch*, is pinned to each allocated GPU.

We assume that all GPUs within a single server node have similar speed and contain the same size memory so that it takes every GPU similar time to finish a batch. The uniformity makes it possible to stream the batches of training data of the same size to all GPU devices using prefetch queue mechanism.

We use the standard TensorFlow prefetch queue to dispatch training data to GPU devices asynchronously. We define a special node to host the prefetch queue and enqueue the tasks of prefetching training data first. Each TensorFlow worker process then dequeues the batches to its corresponding GPU device asynchronously. The prefetch queue is compatible with the graph computation. Our program first defines the queue in the graph and then run the queue operation with a Google TensorFlow session.

On each GPU, a replica is trained by the Momentum Optimization. First a forward inference is performed by the module using the training data. Then the loss for the current replica is calculated. And gradient of variables are calculated.

There are multiple replicas being trained simultaneously so we need to summarize the gradients of all replicas by calculating the mean of gradients in order to update the variables. The summarization and variable update are operated on CPU and they naturally implemented synchronization mechanism of all replicas by waiting all GPUs to finish the processing of a batch of data.

IV. Experiments and Results

A. Data Augmentation

To begin, we wanted to determine the ability of our current neural networks to classify x-ray scattering images with varying levels of noise. We adopted AlexNet [11] as our CNN for image labelling. After training the neural networks with augmented data, we can then analyze how the newly-trained neural networks react to the same x-ray scattering images with varying levels of noise to determine how truly robust our network has become. Fig. 5 compares the performance of our convolutional neural network on four different types of noise: noise-free images, Gaussian noise, Salt and Pepper Noise, and Poisson Counting Statistics. From the graph, we can tell that the precision decreases as more noise is added.

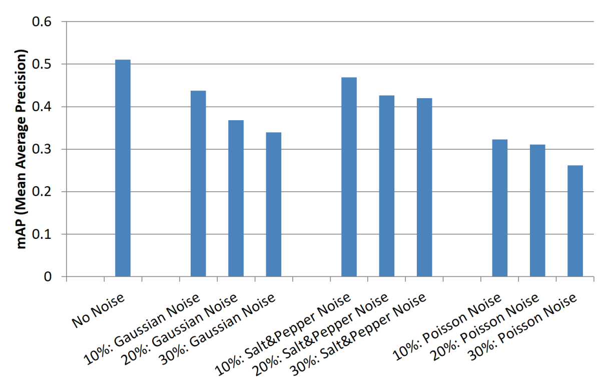

Fig. 5. Performance of CNN without data augmentation on corrupted data.

After training the Neural Networks with augmented data as described in Section III.D, we evaluated their performance by testing the network on the same x-ray scattering images from earlier. As demonstrated in Fig. 6. the CNN's performance before and after training with data augmentation increased significantly. Data augmentation was most successful in improving the network's ability to classify images with Gaussian noise -- the precision increased by about 15% to 18%. The CNN can also now classify images with Salt and Pepper noise with 10% more accuracy and images corrupted with Poisson counting statistic with 8% more accuracy.

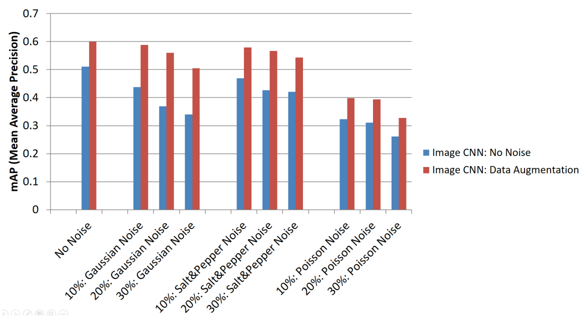

Fig. 6. Comparison of CNN performance with and without data augmentation.

The CNN will tag an x-ray scattering image with 10 distinct known labels. We examined the effects of data augmentation on the network's ability to classify each individual label. Fig. 7 compares the CNN's ability to classify 10 individual labels on images with four different types of noise (a -- noise free; b -- Gaussian; c -- Poisson; d -- salt and pepper). The neural network was able to classify certain labels extremely well (Higher Order, Rings) at a precision of over 80%, and therefore the precision did not increase much after data augmentation. While there were some attributes that were initially hard to classify, data augmentation improved the classification of other labels (Diffuse Lo-q, Diffuse Hi-q, Halo) by 10%. There were also other tags that data augmentation helped very minimally with such as Symmetric Rings and 2-fold symmetry, 4-fold symmetry, 6-fold symmetry. Data augmentation seemed to increase the classification of images with Gaussian noise the most because Gaussian noise had the highest increases in mean average precision (mAP).

In addition, the neural network was trained on images that had all three forms of noise on one image. Fig. 8 describes the impact that a combination of noise had on the mean average precision. Training on combination of noise produced little to no improvement in the performance of the CNN. Since the corrupted in the images had all three forms of noise on the images, the noise may have been too strong for the neural network to learn from the noisy data.

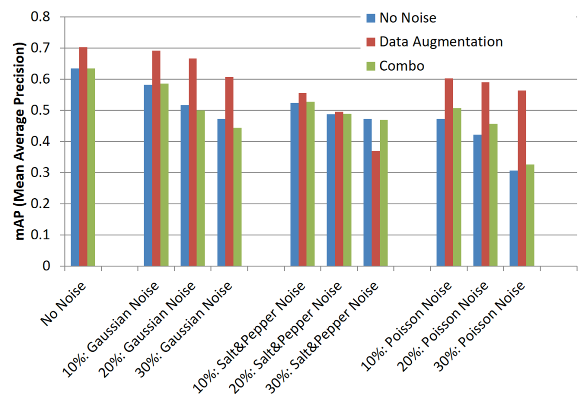

Fig. 8. Performance comparison of CNN trained on noise-free data, data augmentation with single noise types and combined noise augmentation.

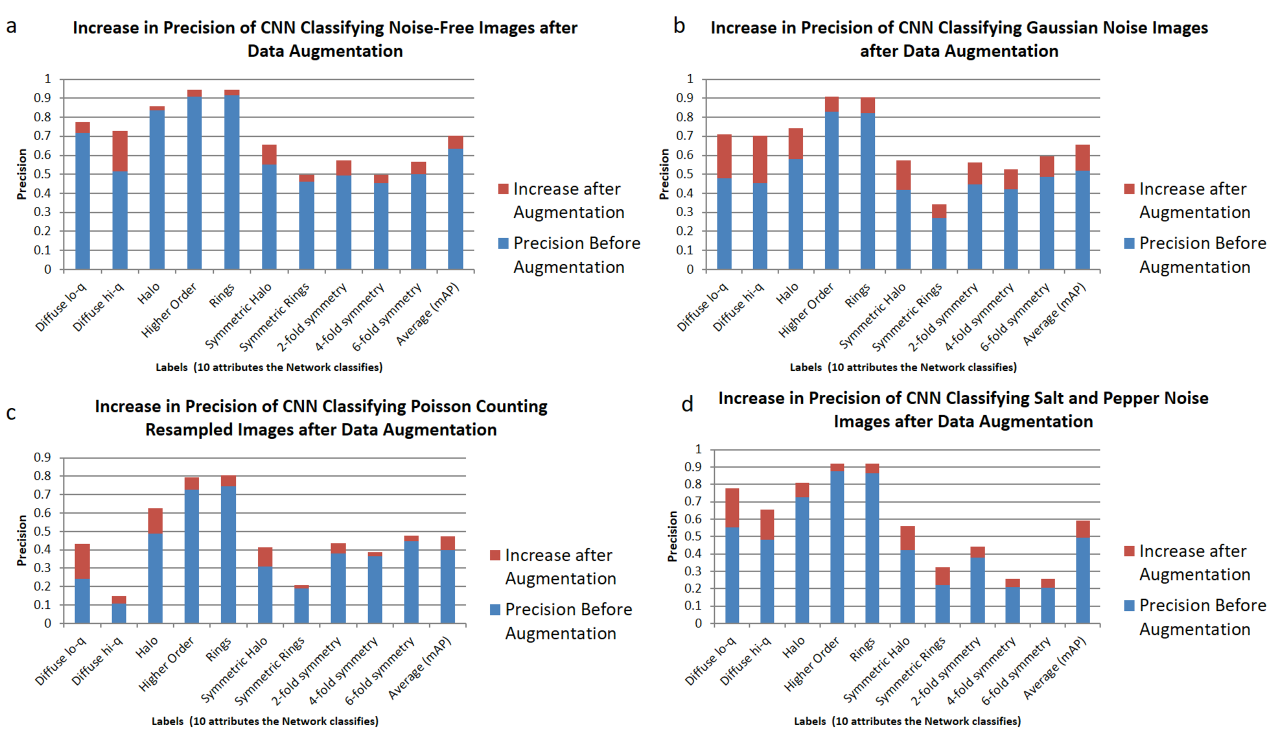

Fig. 7. Performance improvement per label with difference noise types with data augmentation.

B. *Multi-GPU Training*

We deployed the Multi-GPU System Paradigm on a computer cluster at Stony Brook University that contains eight NVIDIA K80 GPUs. We execute the end-to-end training process and train an image CNN module in four different architectures where 0, 1, 4 and 8 GPUs are utilized and compare the total run time and the inference accuracy.

To clarify the comparison, we run the image CNN module training with 3,200 scientific images for 5,000 steps on all four architectures. We display the total runtime and validation result accordingly in Table 1.

TABLE I. TRAINING SPEED COMPARISON ON DIFFERENT MULTI-GPU SETUPS.

Number of GPUs	Total Runtime (s)	mAP
0 (CPU)	27,832	0.944
1	19,500	0.954
4	6,560	0.957
8	4,250	0.951

V. CONCLUSIONS

In this paper, we have evaluated the effects of data augmentation in training CNNs to understand how robust the CNNs are in processing images with noise. We discovered that after training the CNNs with data that contained different amounts and different types of noise, the CNN always performs better than the previous CNN that was trained with only noise-free images. We also found that after training the CNN with augmented data, the performance improved the most for the case of Gaussian noise, suggesting that the data augmentation technique is the most useful when trying to make the neural network robust to Gaussian noise. The network also became more robust for the cases of Salt and Pepper Noise and Poisson Counting Statistics than the network trained with noise-free data. We have shown that data augmentation, specifically noise injection, is a successful regularization technique for training CNNs. We have demonstrated that multiple GPUs are effective to accelerate the training of CNN. In the future, we will investigate how well the neural network performs in classifying real images, rather than synthetic data.

REFERENCES

[1] Wang, Boyu, Kevin Yager, Dantong Yu, and Minh Hoai. 2017. "X-Ray Scattering Image Classification Using Deep Learning." In Applications of Computer Vision (WACV), 2017 IEEE Winter Conference On, 697–704. IEEE.

[2] Wallace, BA. 2000. "Conformational Changes by Synchrotron Radiation Circular Dichroism Spectroscopy." Nature Structural & Molecular Biology 7 (9): 708.

[3] Hajdu, J, KR Acharya, DI Stuart, PJ McLaughlin, D Barford, NG Oikonomakos, H Klein, and LNt Johnson. 1987. "Catalysis in the Crystal: Synchrotron Radiation Studies with Glycogen Phosphorylase B." The EMBO Journal 6 (2): 539.

[4] Anderson, Amy C. 2003. "The Process of Structure-Based Drug Design." Chemistry & Biology 10 (9): 787–97.

[5] Kiapour, M Hadi, Kevin Yager, Alexander C Berg, and Tamara L Berg. 2014. "Materials Discovery: Fine-Grained Classification of X-Ray Scattering Images." In Applications of Computer Vision (WACV), 2014 IEEE Winter Conference On, 933–940. IEEE.

[6] LeCun, Yann, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. 1989. "Backpropagation Applied to Handwritten Zip Code Recognition." Neural Computation 1 (4): 541–551.

[7] Sietsma, Jocelyn, and Robert JF Dow. 1991. "Creating Artificial Neural Networks That Generalize." Neural Networks 4 (1): 67–79.

[8] Zur, R. M., Jiang, Y., Pesce, L. L., & Drukker, K. (2009). Noise injection for training artificial neural networks: A comparison with weight decay and early stopping. *Medical Physics*, *36*(10), 4810–4818.

[9] Goodfellow, Ian, Yoshua Bengio, and Aaron Courville. (2016). Deep Learning. MIT Press.

[10] Perepelitsa, D. V. (2006). Johnson noise and shot noise. *Dept. of Physics, MIT*.

[11] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton. 2012. "Imagenet Classification with Deep Convolutional Neural Networks." In Advances in Neural Information Processing Systems, 1097–1105.