CS229 term project: Data classification for diffraction images

Po-Nan Li*
liponan@stanford.edu
(Dated: June 28, 2018)

We use convolutional neural network to implement a data classifier for diffraction images from serial femtosecond crystallography to identify hit imaging events. The trained network can predict hit events with up to 98% sensitivity and predict miss events with up to 97% specificity on validation data.

I. INTRODUCTION

X-ray free-electron laser (XFEL) facilities, such as Linac Coherent Light Source (LCLS) at the SLAC National Accelerator Laboratory, have enable new ways of studying structures of biological cells or nano-particles and their dynamics. The light source illuminates a sample with up to 10^{12} photons in a single ultrafast pulse, providing both high temporal and high spatial resolution imaging. Combined with diffraction imaging techniques such as serial femtosecond crystallography (SFX) [1] or coherent diffraction imaging (CDI) [2], such powerful light sources recently have been employed to study the Å-scale structure and dynamics of proteins, which are otherwise elusive.

LCLS currently operates at a repetition rate of 120 Hz, i.e., data is generated at 120 snapshots per second from a megapixel camera, generating a huge volume of data (many TBs per experiment). While such "big data" are potentially very informative, the transfer and storage can be costly, and among them a considerable portion of images, up to 80%, doesn't contain signal from the sample; in other words they are "misses," like the one shown in FIG. 2(a). It is therefore desirable to have an efficient approach that can identify and categorize hit events in an online fashion.

To address these problems, here we develop a framework for rapid data classification. Naively, "hit" and "miss" images can be distinguished by measuring the total intensity recorded in a single image; however such approach can be vulnerable if the background scattering is strong and/or the signal is faint. This project aims to employ machine learning methods to look into a large volume of LCLS image data, and classify them as "hit" or "miss". We capitalizes on the "big data" nature of LCLS experiments, which generate millions of images, to train and test our algorithm. The impact of rapid and accurate image classification and interpretation is highly significant: fast online and offline experimental data analysis will reduce the need of data transfer and storage, and improve the data quality.

Here we train a 18-layer convolutional neural network (CNN) on experimentally measured free-electron laser diffraction images of protein crystals to classify hit or miss. Since hit events are relatively rare in the whole volume of data, we will particularly emphasize the true positive rate, a.k.a. sensitivity. Besides, we will also discuss methods for preprocessing data so the presented framework can be extend to different samples and experiment settings.

* Current address: Department of Electrical Engineering, Stanford University, CA, USA

II. RELATED WORK

Due to the big-volume nature of imaging experiments, automated clustering algorithms are always desired. Yoon *et al.*[4] and Duan *et al.*[3] have have demonstrated the application of principal component analysis (PCA) for single particle imaging (SPI) and for transmission X-ray microscopy (TXM), respectively. Tokuhisa *et al.*also studied a classification pipeline that analyzes images' similarity [5].

Though deep learning and CNN [6] have been extensively utilized for literally every aspect of life, little has been done for applying them to XFEL imaging. Yann *et al.*applied CNN for classification of protein, in real space but not spectral space [10]. Wang *et al.*trained CNN on simulated diffraction images, but didn't discuss the application for crystallographic data.

III. DATASET AND FEATURES

A. Diffraction images

Here we look into the serial femtosecond crystallography (SFX) imaging data of streptavidin, a extensively used and studied protein in biology [8]. FIG. 1 shows the reconstructed structure in a cartoon fashion. The experiment captured the diffraction patterns of crystallized streptavidin in 10 sequential experiment "runs," numbered 90 − 96 and 98 − 100, 97 being an empty run. Depending on the time period the run was executed, a run is consisting of up to ten thousands of event, each event including a 1750×1750 2D gray-scale image. If an event was a hit, its image should record several Bragg peaks, which can be used to calculate the corresponding crystal structure. This project aims to identify the hit images, or equivalently images with Bragg peaks. To speed up training and testing and reduce the need of memory, we crop images to its central 350 × 350 region, where Bragg peaks (if any) are preserved. Note here we only crop the images, rather than downsample, in order to preserve Brag peaks, which are typically one-pixel big. FIG. 2(b) shows an example of a preprocessed hit image, where several hot pixels can been seen.

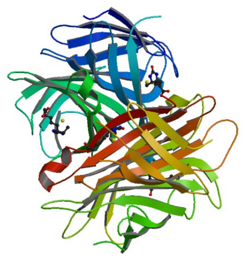


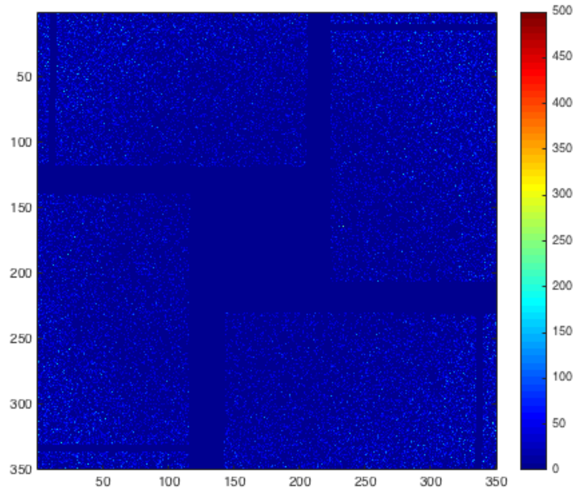
FIG. 1: Cartoon of streptavidin complex.

B. Labeling

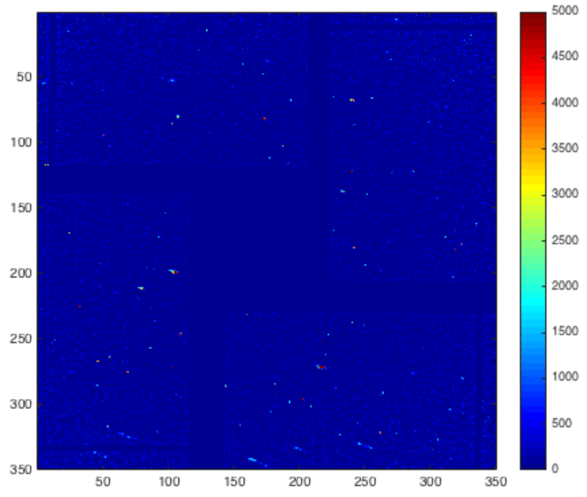
Technically there is no formal definition for hit or miss, as the diffraction pattern is elusive until it is reconstructed to a real-space image. In order to label the data, we use an algorithm to find Bragg peaks in each image, and if the number of found peaks is larger than a given threshold then we say it's a hit, otherwise a miss. FIG. 3 exhibits the labels of run 90, where events whose number of peaks is below 2 is marked as miss, above 10 is marked as hit; others are dropped from training and testing data, as in practice they ambiguously can be hit or noisy miss events.

IV. METHOD

In this project we implement and train a convolutional neutral network (CNN) to perform data classification. While there are several candidates out there, such as linear regression, support vector machine (SVM) and clustering [3], we choose CNN because it is potentially capable of revealing "hidden science" behind the diffraction images and is shift-invariant to the features on the image. Specifically, CNN might be able to discover structures that only exist in hit images, such as faint Bragg peaks buried in the noise. Here we employ a multi-layer CNN implemented by using Keras [12] with TensorFlow [13] as backend, which consists multiple 2-D convolution, max-pooling, dropout, batch normalization, fully-connected layers, and an output layer. For example, FIG. 4 visualizes the four filters in the first convolution layer. Table I shows the full architecture of our CNN model. Dropout layers are used to prevent over-fitting; sample-wise (layers 3 and 7) and feature-wise batch-normalization layers are also implemented to regulate value range of data. We use cross-entropy as loss function, and stochastic gradi-



(a) A 'miss" event



(b) A "hit" event

FIG. 2: Example of a (a) miss and (b) hit event. Note (b) contains several pixel-big Bragg spots, which have high pixel value.

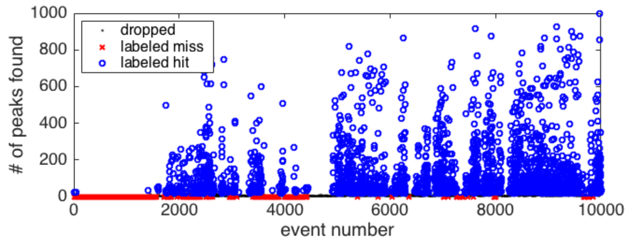


FIG. 3: Number of peaks vs. event index in run 90. Blue circles mark the events with hit labels; red crosses mark the events with miss labels. Others are dropped from training and testing as they are practically ambiguous.

TABLE I: Architecture of our CNN. Conv: 2-D convolution layer. Drop: dropout. BN: batch normalization. ReLU: rectified linear unit. Max: max-pooling. Flat: flatten. FC: fully-connected.

Layer	Type	Output shape	Filter size	# params
0	Input	$350 \times 350 \times 1$	-	0
1	Conv	$344 \times 344 \times 4$	7×7	200
2	Drop	$344 \times 344 \times 4$	-	0
3	BN	$344 \times 344 \times 4$	-	688
4	ReLU	$344 \times 344 \times 4$	-	0
5	Max	$68 \times 68 \times 4$	-	0
6	Conv	$62 \times 62 \times 4$	7×7	788
7	BN	$62 \times 62 \times 4$	-	124
8	ReLU	$62 \times 62 \times 4$	-	0
9	Max	$12 \times 12 \times 4$	-	0
10	Drop	$12 \times 12 \times 4$	-	0
11	Flat	576	-	0
12	FC	16	-	9232
13	BN	16	-	32
14	ReLU	16	-	0
15	Drop	16	-	0
16	FC	16	-	34
17	ReLU	16	-	0

ent descent (SGD) as optimizer, with parameters learning rate 0.002, learning rate decay 0.0001, momentum 0.9, and batch size 100.

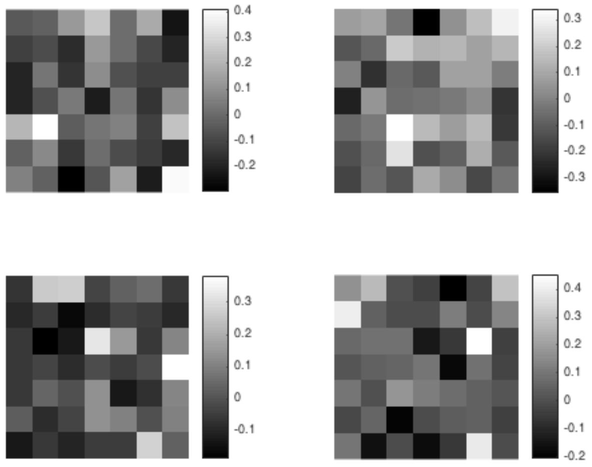


FIG. 4: Trained filters in the first 2-D convolution layer.

V. RESULTS

A. Cross-run validation

We validate our trained CNN by predicting on 10 experiment runs, including 3 runs that have been used for training, and other 7 runs that CNN had never seen. Table II shows the prediction results, where accuracy is defined as the percentage of events that are correctly classified, and the definitions of sensitivity (sen.) and specificity (spe.) will be discussed in the following paragraph.

TABLE II: Prediction results on training data (runs 90 – 93) and validation data (runs 94 –96 and 98 – 100). Sen: sensitivity. Spe: specificity. Acc: accuracy.

Run	# events	% hit	Sen	Spe	Acc
90	11429	0.3366	0.9974	0.8685	0.9119
91	4755	0.3171	1.0000	0.9593	0.9722
92	14244	0.0791	0.9663	0.9442	0.9459
93	5820	0.2864	0.9754	0.9839	0.9814
94	13820	0.3727	0.8794	0.9451	0.9206
95	12647	0.1468	0.9822	0.9728	0.9741
96	12157	0.3822	0.9884	0.9113	0.9408
98	14208	0.0934	0.8734	0.6071	0.6320
99	11081	0.8471	0.9892	0.1488	0.8608
100	10730	0.4781	0.9004	0.5705	0.7282

B. Confusion matrix

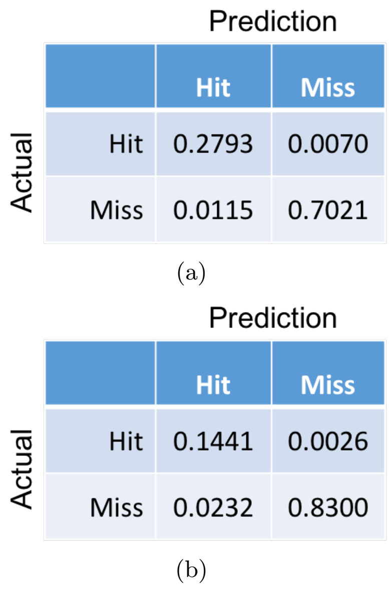
In a real setting of XFEL imaging, despite the high frequency operation, a large portion of events doesn't contain signal from the sample. As as result, we should emphasize the importance of sensitivity of the prediction, defined as $TP/(TP + FN)$, where TP and FN are the percentages of true positive and false negative, respectively. On the other hand, we don't want to classify a hit event, which is very rare and valuable, as miss. So specificity, defined as $TN/(TN+FP)$, where TN and FP are true negative and false positive rates, is also expected to be high. Further, we can analyze the confusion matrix of the prediction. FIG. 5(a) shows the confusion matrix on run 93, which is a subset of training data. FIG. 5(b) shows the confusion matrix on run 95, which is a subset of testing data.

C. Cross-experiment validation

To test the robustness of the trained CNN, we also use it to predict on data from other XFEL experiments. Unfortunately the CNN fails. Possible reasons are discussed in the following section.

D. Computation time

Our CNN takes roughly 12 ms to predict an cropped image, whereas a peak finding algorithm which is currently being used at LCLS takes roughly 1.4 s to find the peaks on the same image. For the purpose of identifying



(a)

		Prediction	
		Hit	Miss
Actual	Hit	0.2793	0.0070
	Miss	0.0115	0.7021

(b)

		Prediction	
		Hit	Miss
Actual	Hit	0.1441	0.0026
	Miss	0.0232	0.8300

FIG. 5: Confusion matrix of the CNN on (a) run 93 and (b) run 95.

hit events, our CNN outperforms traditional method by a 100-fold faster speed.

VI. DISCUSSION

While in general our CNN achieves very high classification accuracy, we should point out that the false positive rate of prediction is generally higher than the false negative rate, which suggests that a portion of miss events is incorrectly identified as hit. This might be due the nature of noisy diffraction images, which contain random pepper and salt pixels. Besides, the CNN didn't work well with a few runs, e.g. runs 98, 99 and 100, where the prediction accuracy drops below 90%. This might be explained by the fact that each run has distinct experiment condition, e.g. flux level and sample concentration. We are now working on a better data preprocessing pipeline to eliminate such discrepancies across different runs or experiments.

In addition, we notice that, as a rule of thumb, with the data size currently available to us, the total number of parameters of the CNN should be less than 10 thousands; otherwise it would be more difficult to train. While compared with practical image recolonization problems, binary classification is relatively easy, we found the CNN needs at least three convolutional layers to be useful. Utilization of published CNN architectures such as GoogleNet, AlexNet or VGG16 is currently under investigation.

VII. CONCLUSION

We implemented and trained a CNN that can classify diffraction images into miss or hit. Our framework achieves both high sensitivity for hit events and high specificity for miss events. The developed approach might be utilized at an X-ray imaging beambine in an on-line fashion to reduce the burden of data analysis and transfer.

VIII. ACKNOWLEDGMENT

This research project was jointly supervised by SLAC scientists Dr. Henry van den Bedem and Dr. Chun Hong Yoon. I am grateful for their mentorship, and I also thank them for providing access to large experimental datasets and computing resources at SLAC.

[1] J. Tenboer *et al.*, Science **324**, 1246 (2014).
[2] J. Miao, P. Charalambous, J. Kirz, and D. Sayre, Nature **400**, 342 (1999).
[3] X. Duan *et al.*, Sci. Rep. **6**, 34406 (2016).
[4] C. H. Yoon *et al.*, Opt. Exp. **19**, 16542 (2011).
[5] A. Tokuhisa *et al.*, J. Synchrotron. Rad. **20**, 899 (2013).
[6] Y. LeCun, L. Bottou, Y. Bengio and P. Haffner, Proc. IEEE **86** 2278 (1998).
[7] C. H. Yoon *et al.*., Sci. Rep. **6**, 24791 (2016).
[8] M. S. Hunter *et al.*, Nature. Comm. **7**, 13388 (2016).
[9] A. Berntson, V. Stojanoff, and H. Takai, J. Synchrotron Rad. **10**, 445449 (2003).
[10] M. L.-J. Yann and Y. Tang, Proc. AAAI, 1373 (2016)
[11] B. Wang, K. Yager, D. Yu, and M. Hoai, arXiv:1611.03313 (2016).
[12] Chollet and François, Software available from `https://github.com/fchollet/keras` (2015).
[13] M. Abadi *et al.*, Software available from `http://tensorflow.org` (2015).