AI for Science: Use-Cases and Lessons

Zhengchun Liu

Assistant Computer Scientist at Data Science and Learning Division

April 15th, 2020
CELS Coffee Talk

About me and my work

- Assistant Computer Scientist at Data Science and Learning Division
- High Performance Wide Area Data Transfer, logs mining, characteristic and optimization
- Data science and machine learning for computing system, e.g., performance modeling, bottleneck detection and reasoning.
- AI for Science, e.g., X-ray at APS, Climate Simulation, Accelerator at APS etc.
- Looking for collaboration on applying AI for more other domains.

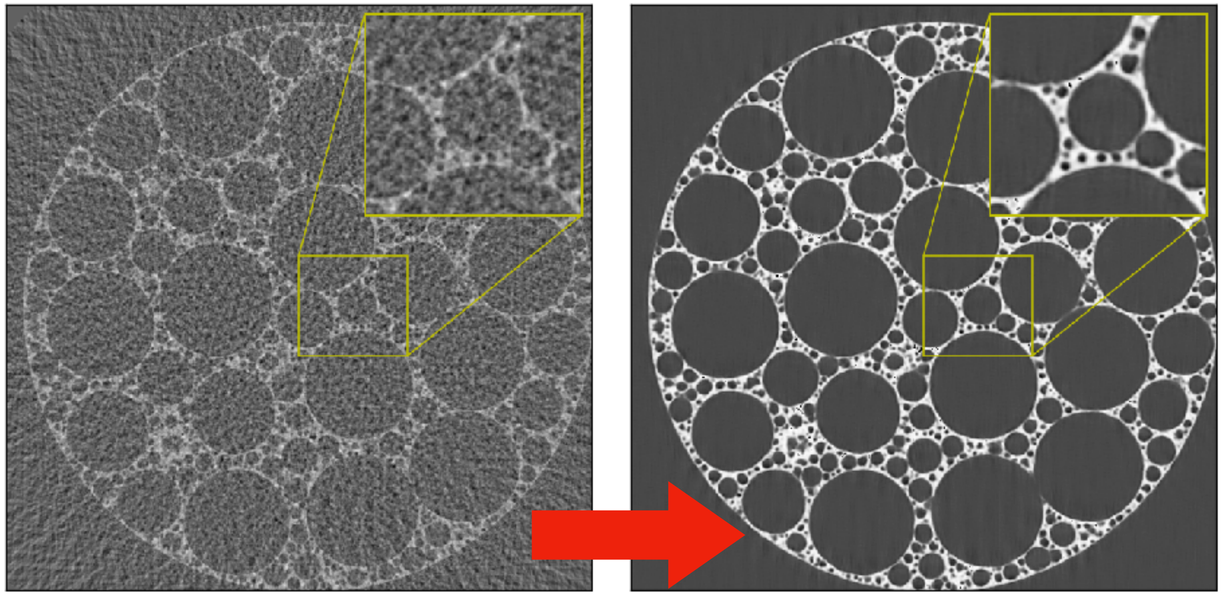
Advancing X-ray Tomography using AI (TomoGAN)

Motivation

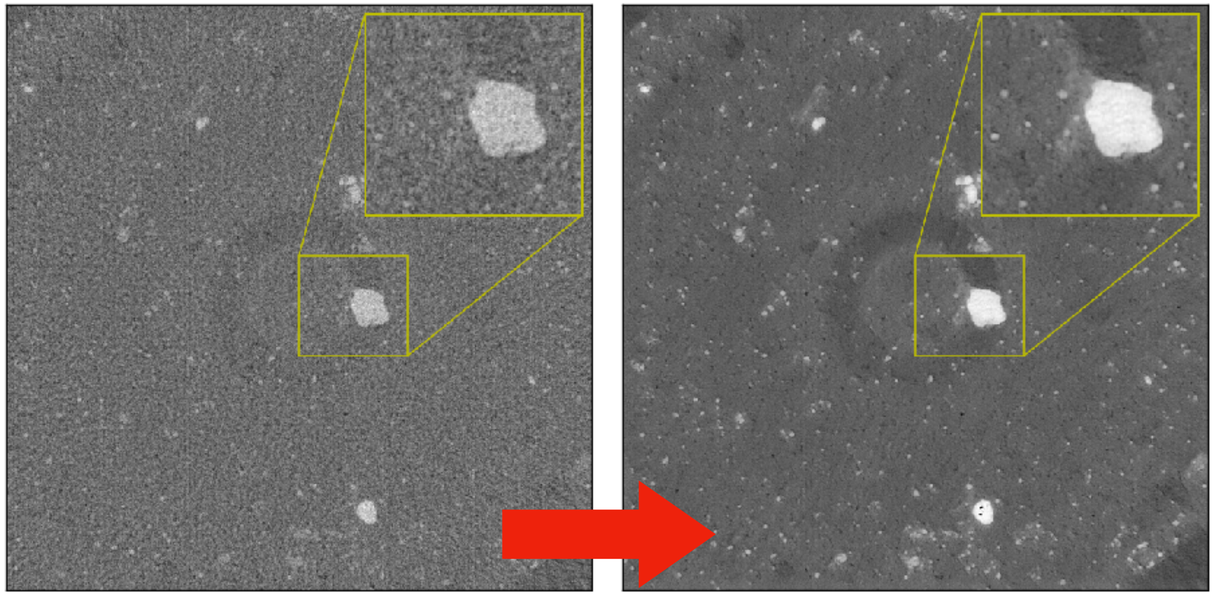
(1) lower X-ray dosage for sensitive sample like bio-sample;

(2) faster experiment to capture dynamic features, like in fast chemical reaction processes;

(3) smaller dataset and less computation for [near] realtime tomography imaging.



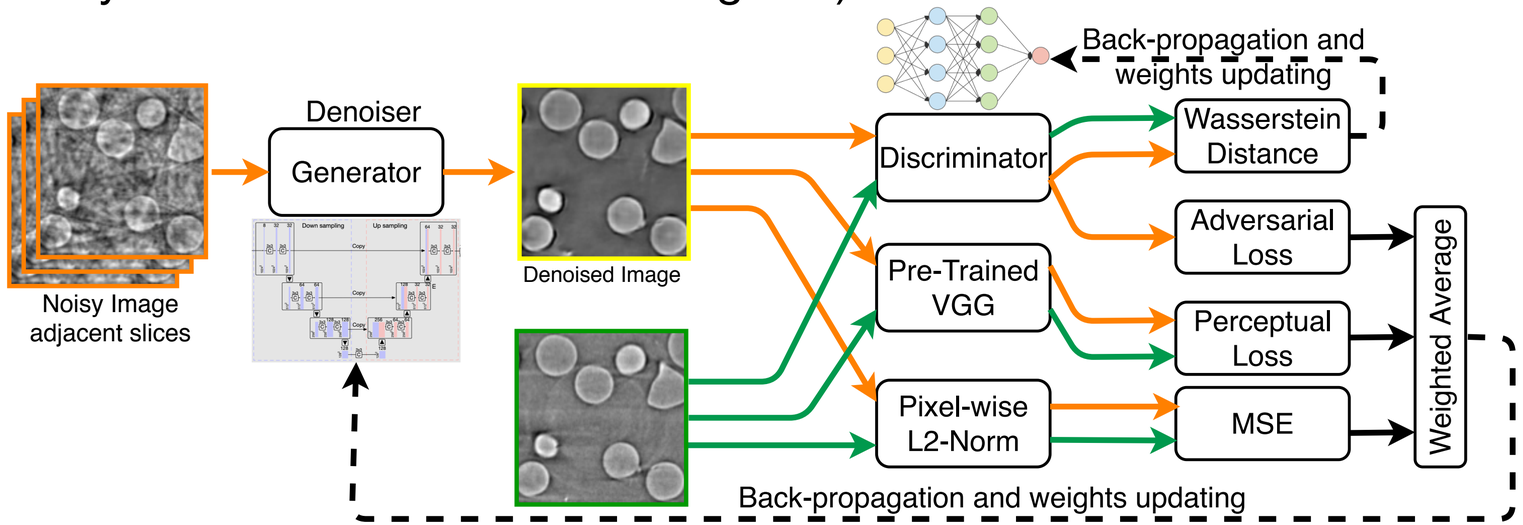
On the left, the results of conventional reconstruction, which are highly noisy. On the right, those same results after denoising with TomoGAN.



Model is trained with one shale sample imaged at APS and tested with **another** shale sample imaged at Swiss Light Source (SLS).

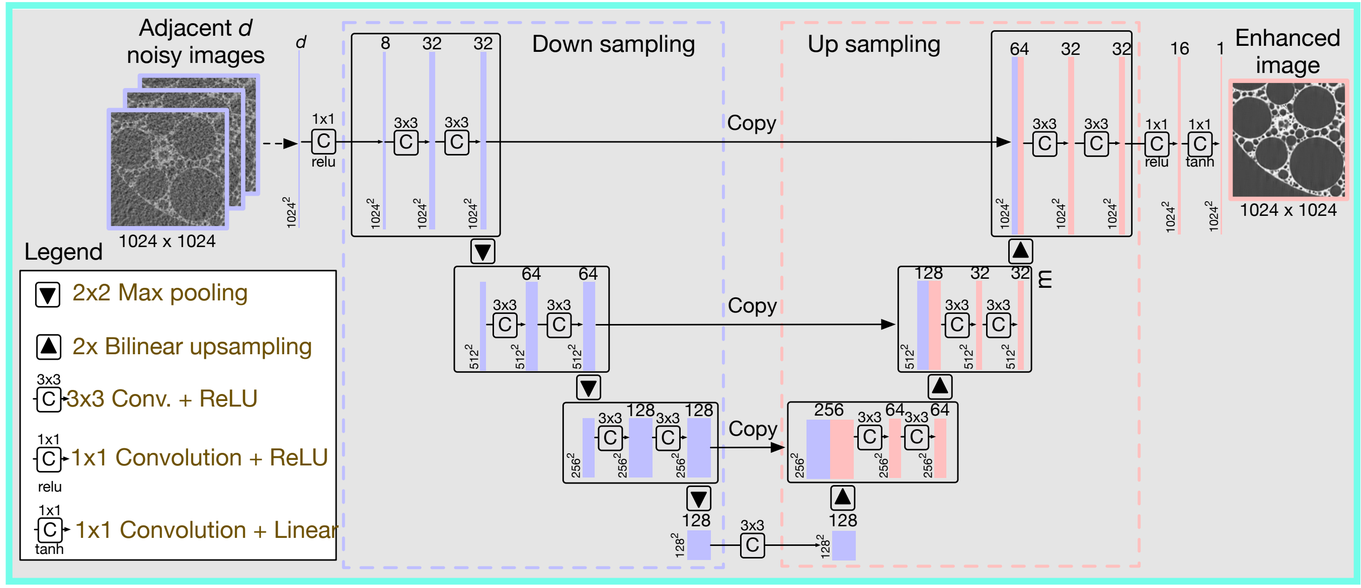
Method

A generative adversarial network (GAN) is a class of machine learning systems in which two neural networks, generator (G) and discriminator (D), contest with each other in a game (in the sense of game theory, often but not always in the form of a zero-sum game).



In our model, the discriminator's job remains unchanged, but the generator is tasked not only with fooling (indistinguishable) the discriminator but also with being near the ground truth output in an L2 sense.

The discriminator works as a helper to train the generator that we need to denoise images.



Our Generator Architecture

Experiments

Datasets

- **Three foam simulation datasets, each with 1024 slices**
- **Two shale samples imaged at both APS and SLS, totals four datasets and each with 2048 slices.**

Label	projection	reconstruction	Facility	Sample	Scan	Axis
tomo_00001	(1501, 1792, 2048)	(1792, 2048, 2048)	APS	B1	hornby	1024
tomo_00002	(1501, 1792, 2048)	(1792, 2048, 2048)	APS	N1	blakely	1029
tomo_00003	(1441, 2048, 2048)	(2048, 2048, 2048)	SLS	B1	hornby	1011
tomo_00004	(1441, 2048, 2048)	(2048, 2048, 2048)	SLS	N1	blakely	1048

Low dose cases

- **Sparse views**

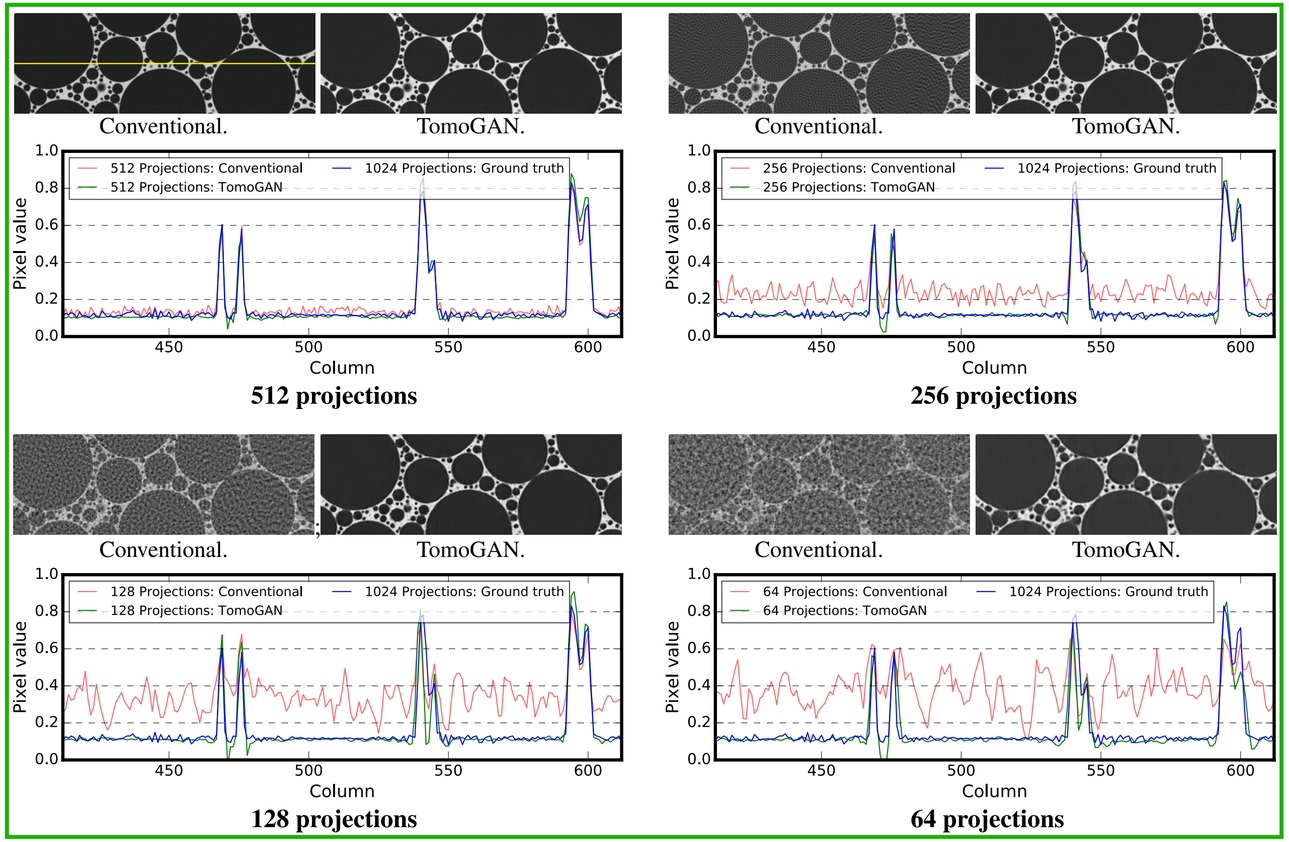
  Subsample the original, (i.e., normal dose) projections to 1/2, 1/4, 1/8 and 1/16 for experiments and model evaluation.

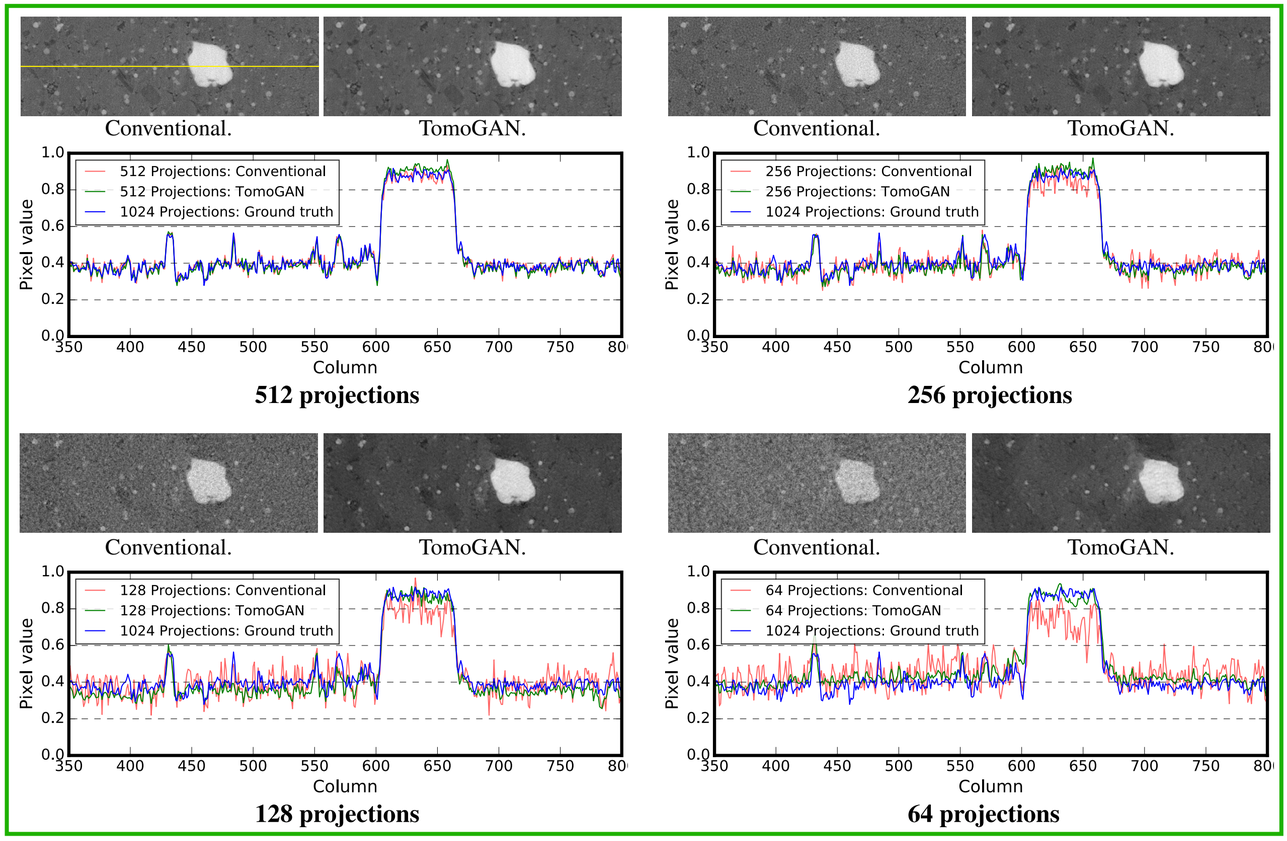
- **Short exposure time.**

  For simulation datasets, we simulate x-ray projections with different photon intensities to simulate different exposure times

  For experimental shale datasets, we used added noise using a Poisson distribution to simulate different exposure times.

Results - Sparse views





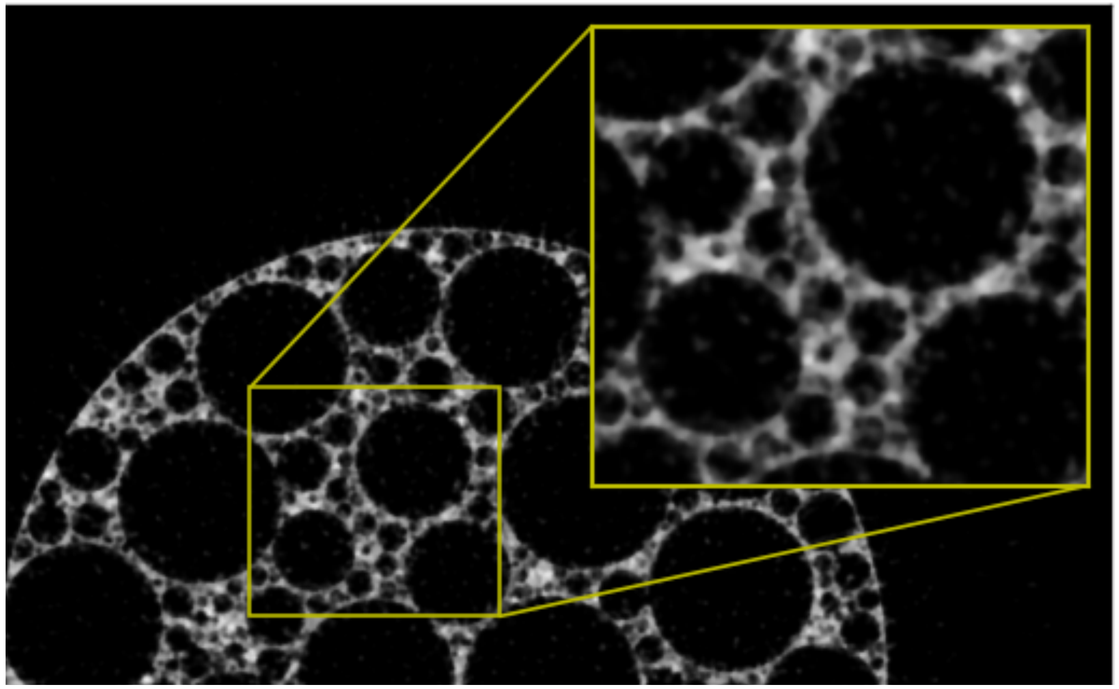
Conventional vs. TomoGAN-enhanced reconstructions of simulated (left) data and shale (right), subsampled to (512, 256, 128, 64) projections. In each group of three elements, the two images show conventional and TomoGAN reconstructions, while the plot shows conventional, TomoGAN, and ground truth values for the 200 pixels on the horizontal line in the top left image.

Computational superiority

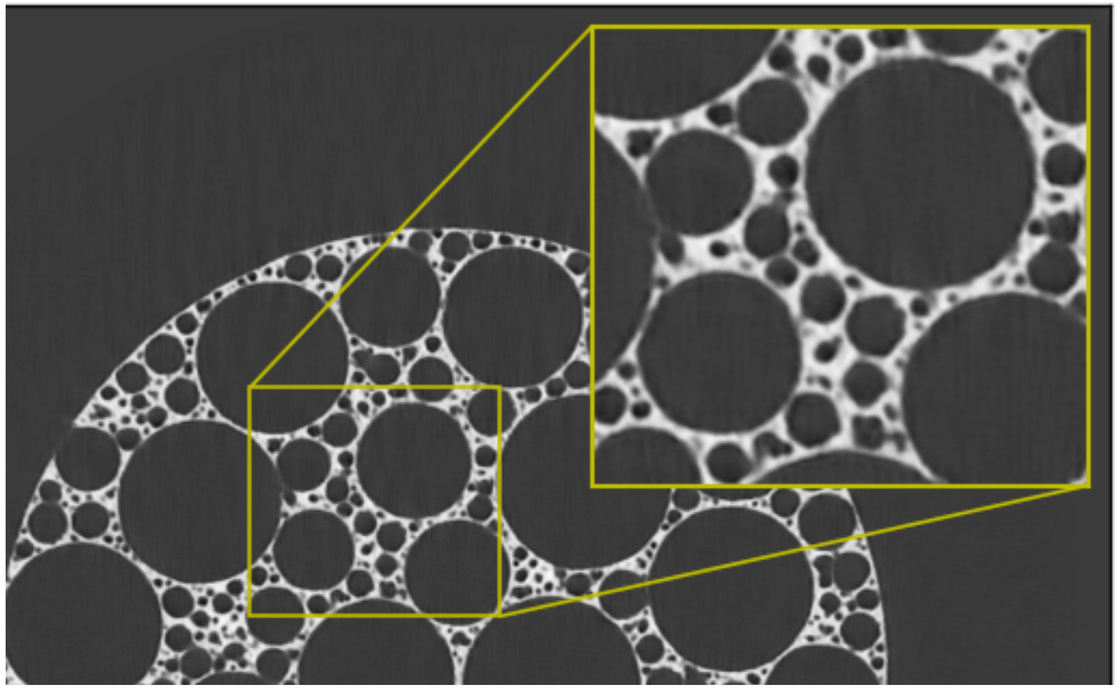
The filtered back projection (FBP) algorithm takes 40 ms to reconstruct one image (using TomoPy) and TomoGAN takes 30 ms to enhance the reconstruction, totals **70 ms** per image.

In contrast, the SIRT based solution (using TomoPy) takes **550 ms** (400 iterations), i.e., 8x faster. Times are measured using one Tesla V100 graphic card.

Moreover, iterative reconstruction does not provide better image quality than does our method.



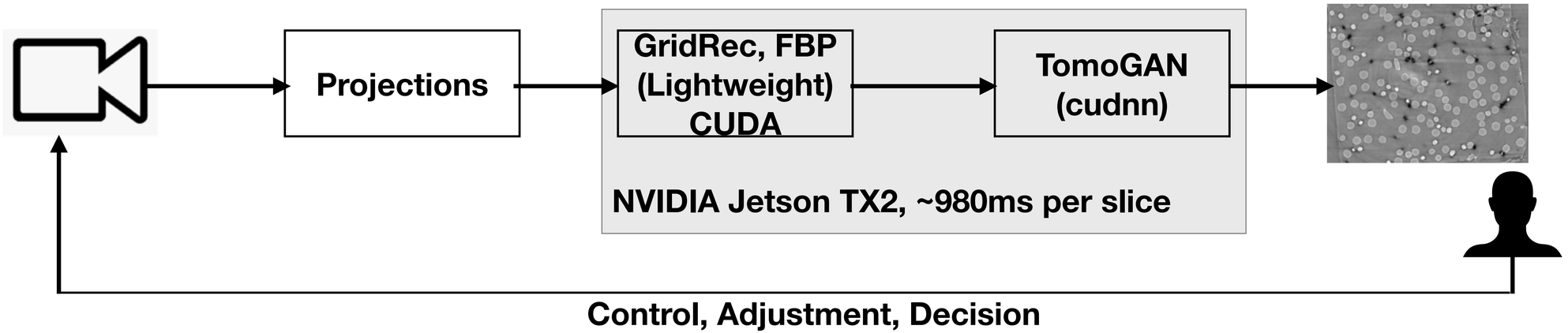
SIRT + total variation postprocess.



Filtered back projection + TomoGAN post-process.

TomoGAN - Tomography at Edge

- ☐ **Both Tomography and DL are computation intensive but both GPU typically helps a lot;**
- ☑ **A GPU friendly tomography for a rough (noisy) results plus DL based enhancement;**
- ☐ **Fusion of analytical (human knowledge) and deep learning (data driven).**



~ $300, maximum 15 watts

With **Viktor Nikitin et al.**

Make it usable

Hack and Play

open source implementation, better to have a GPU for training

```
Git clone git@github.com:ramsesproject/TomoGAN.git

python ./train.py -ld noise-img.hdf5 -nd clean-img.hdf5

python ./infer.py -ld ld-prod.hdf5
```

X as a Service

DLHub

Data and Learning Hub for Science

B. Blaiszik. arXiv:1811.11213

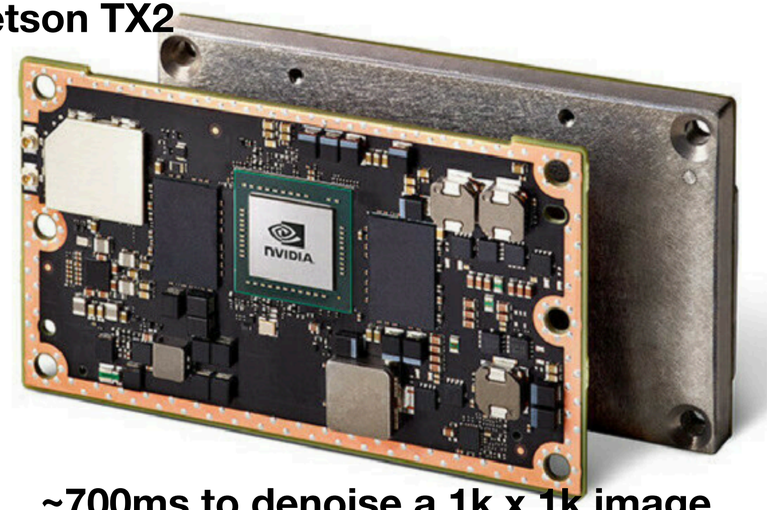
```
from dlhub_sdk.client import DLHubClient
dlhub = DLHubClient()

model = dlhub.get_id_by_name("tomoGAN")
data  = h5py.File("tomo_ld.hdf5", "r")["ld_img"]
pred  = dl.run(model, data)
```

Plug and Play

Abeykoon et al.

Edge TPU

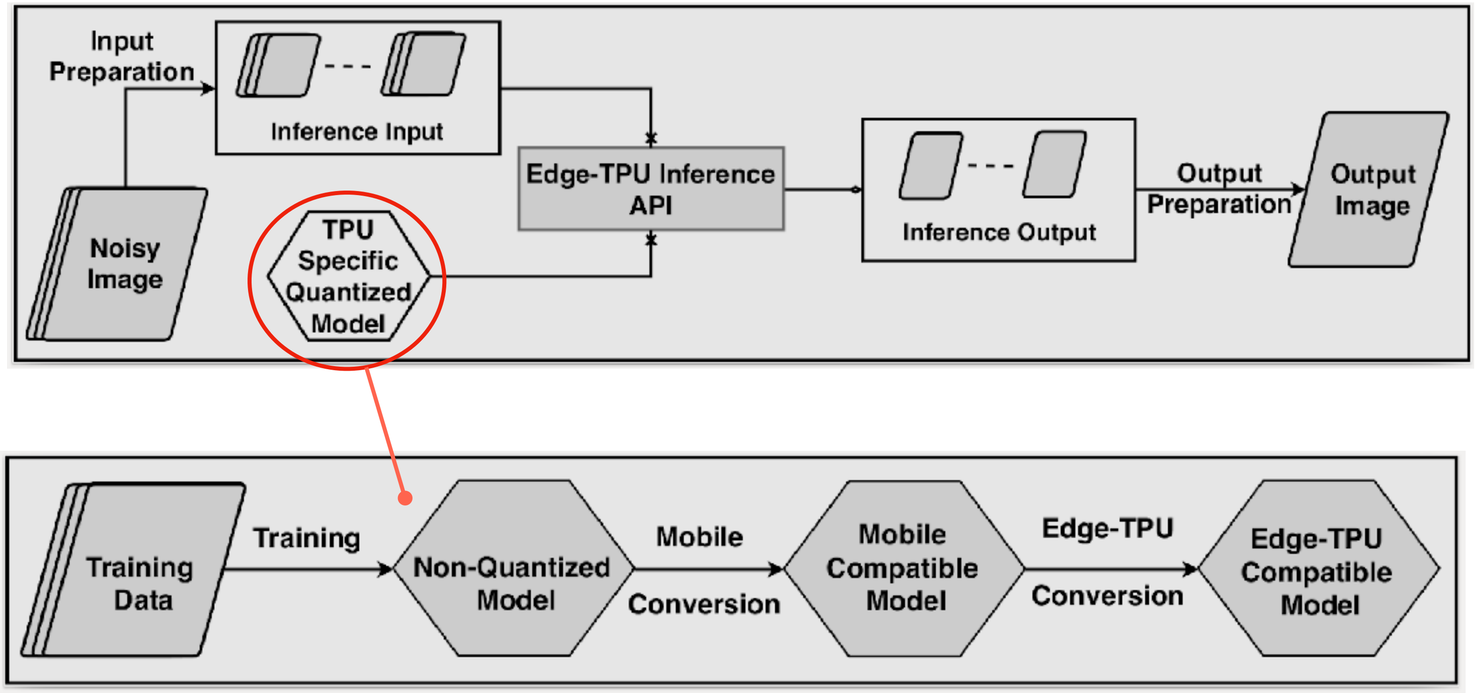
Jetson TX2



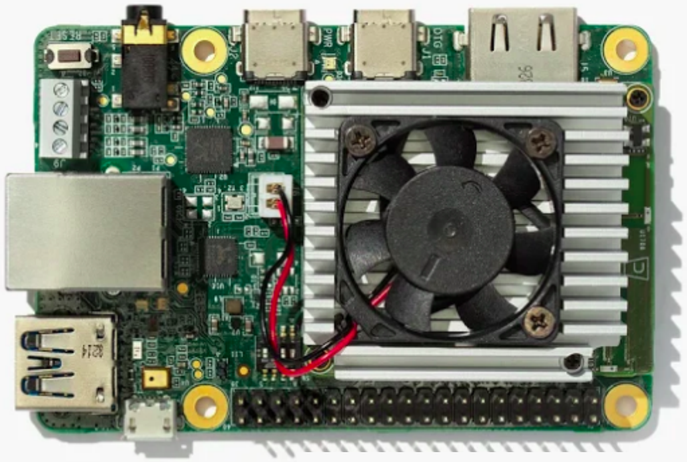
~700ms to denoise a 1k x 1k image

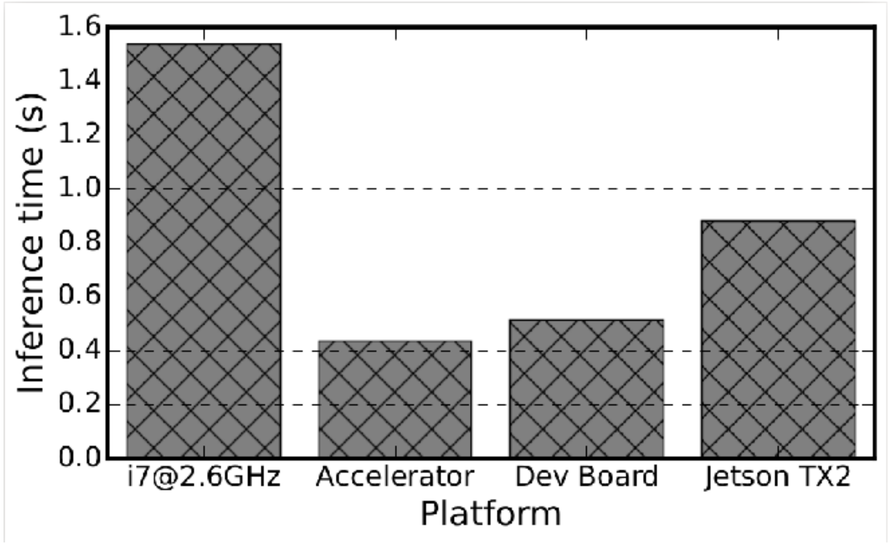
python: Tensorflow and Keras based;
C++ : DNNL(MKL-DNN) based, good for CPU based e.g., KNL;
C++, CUDA: cuDNN and cuda based, good for NVIDIA GPU;
Pytorch: upon request

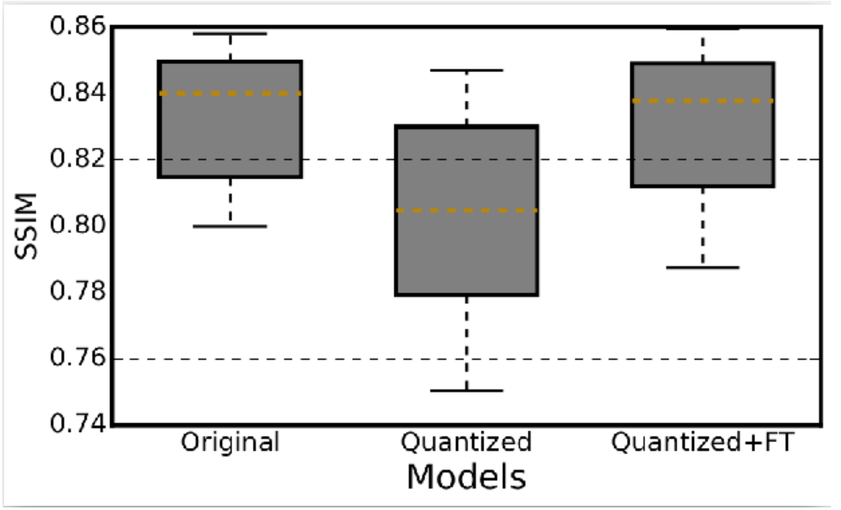
Make it usable - Continue Details



TPU Dev Board







TPU Accelerator

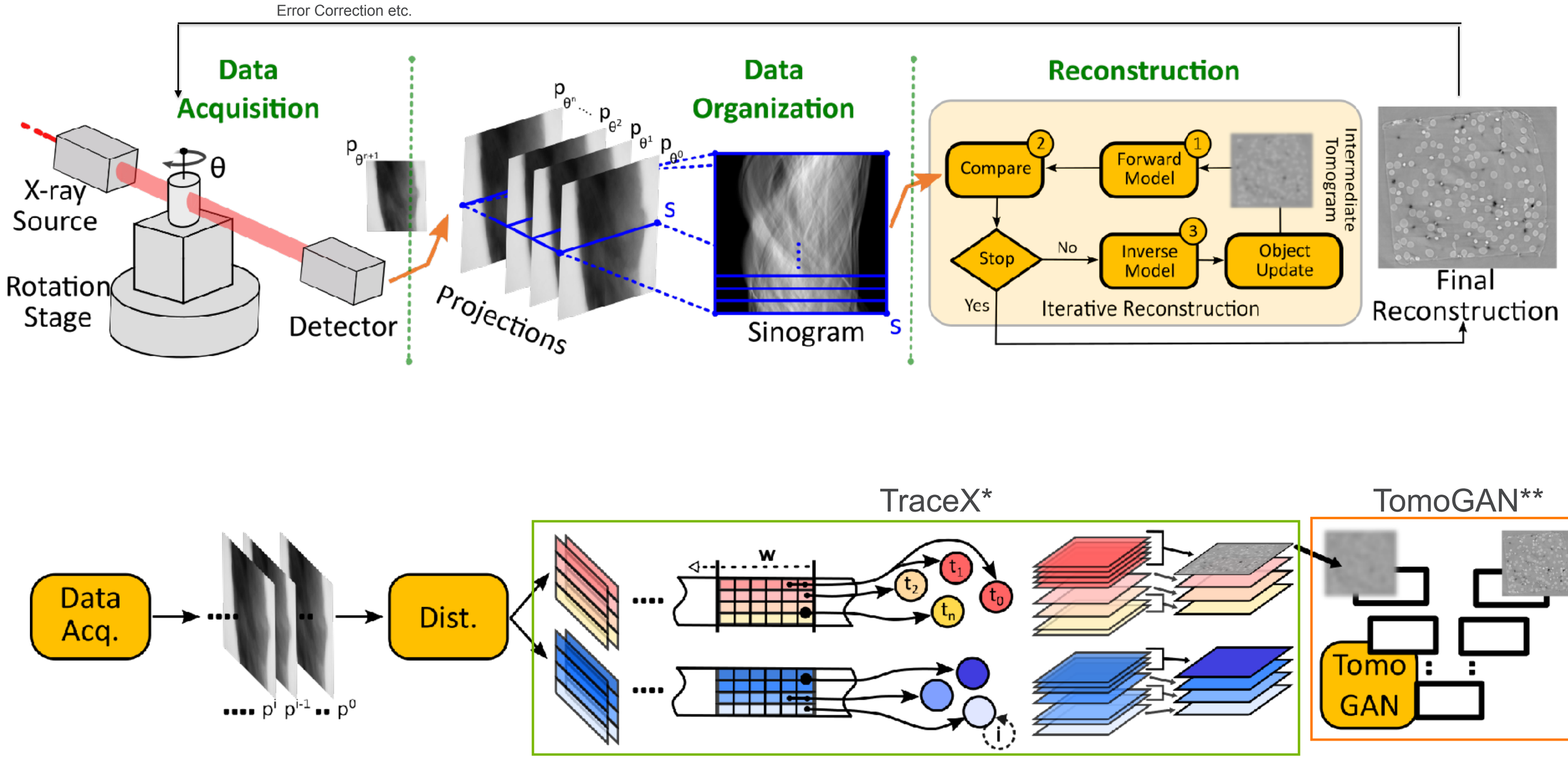
Deep Learning Accelerated Light Source Experiments

An Extended use of TomoGAN

Motivation

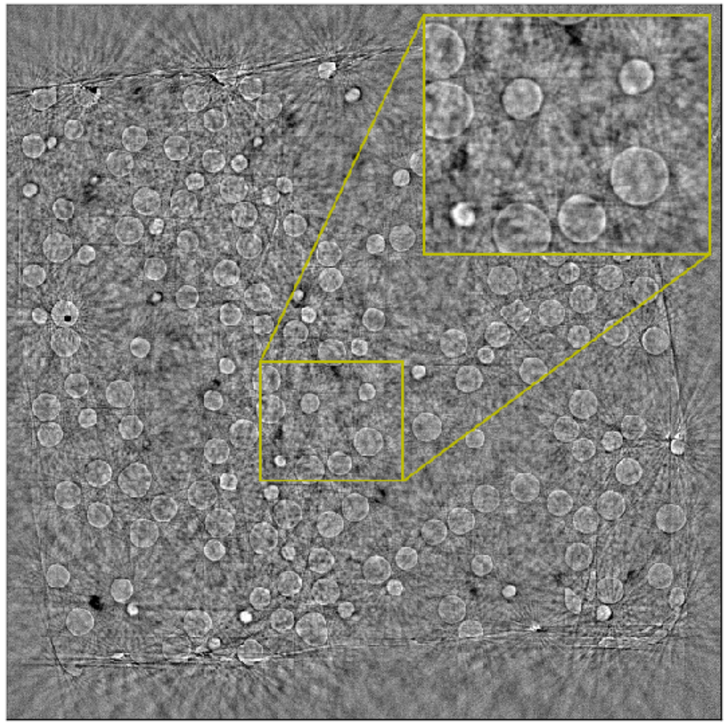
It enables:

- Error Correction early in experiments;
- Experiment Steering, e.g., early stoping;
- Detection of features in hierarchical structures;
- Change data acquisition to capture dynamic features;
- Adjust experimental parameters on the fly;

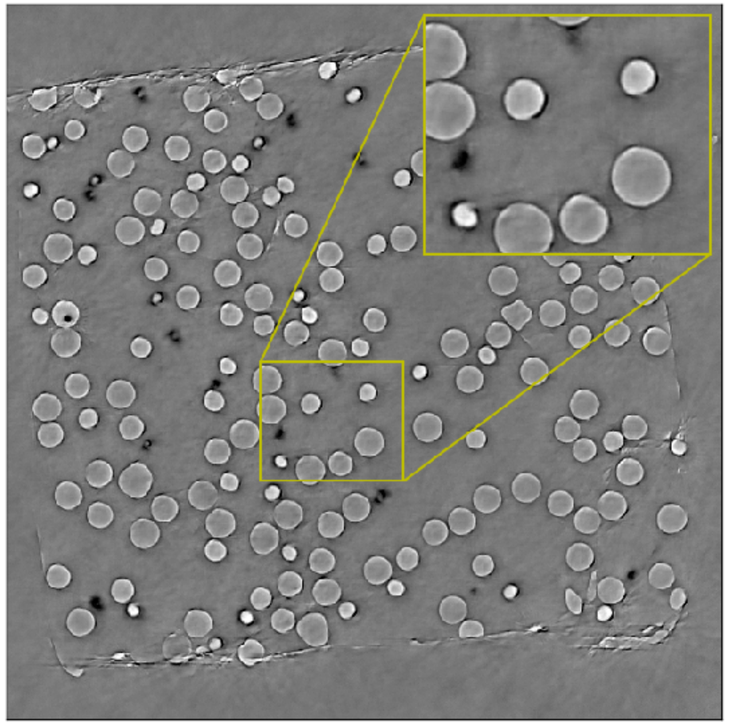


* T. Bicer. Advanced structural and chemical imaging. 2017

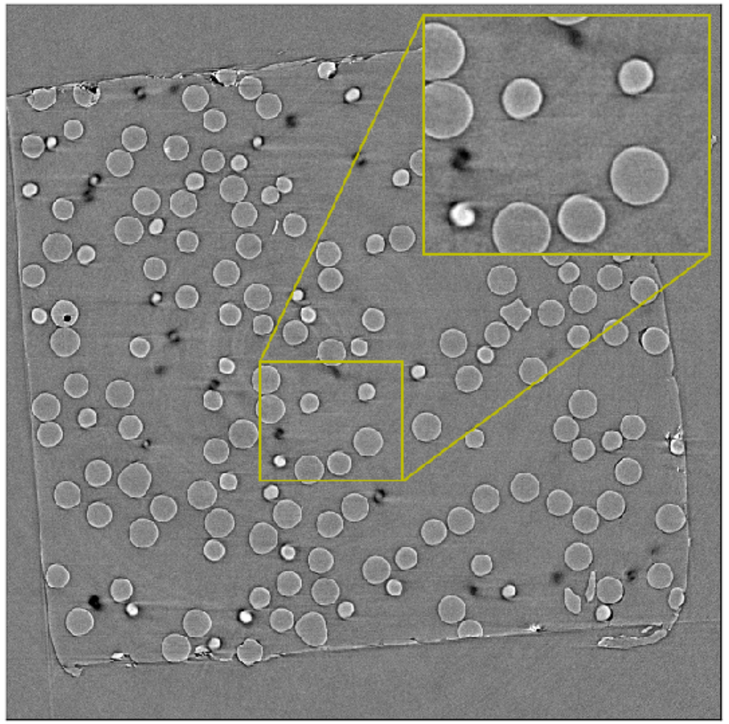
** Z. Liu. TomoGAN. arXiv:1902.07582



with data up to 462s (480 projections), before enhancement;



with the same data, after enhancement;

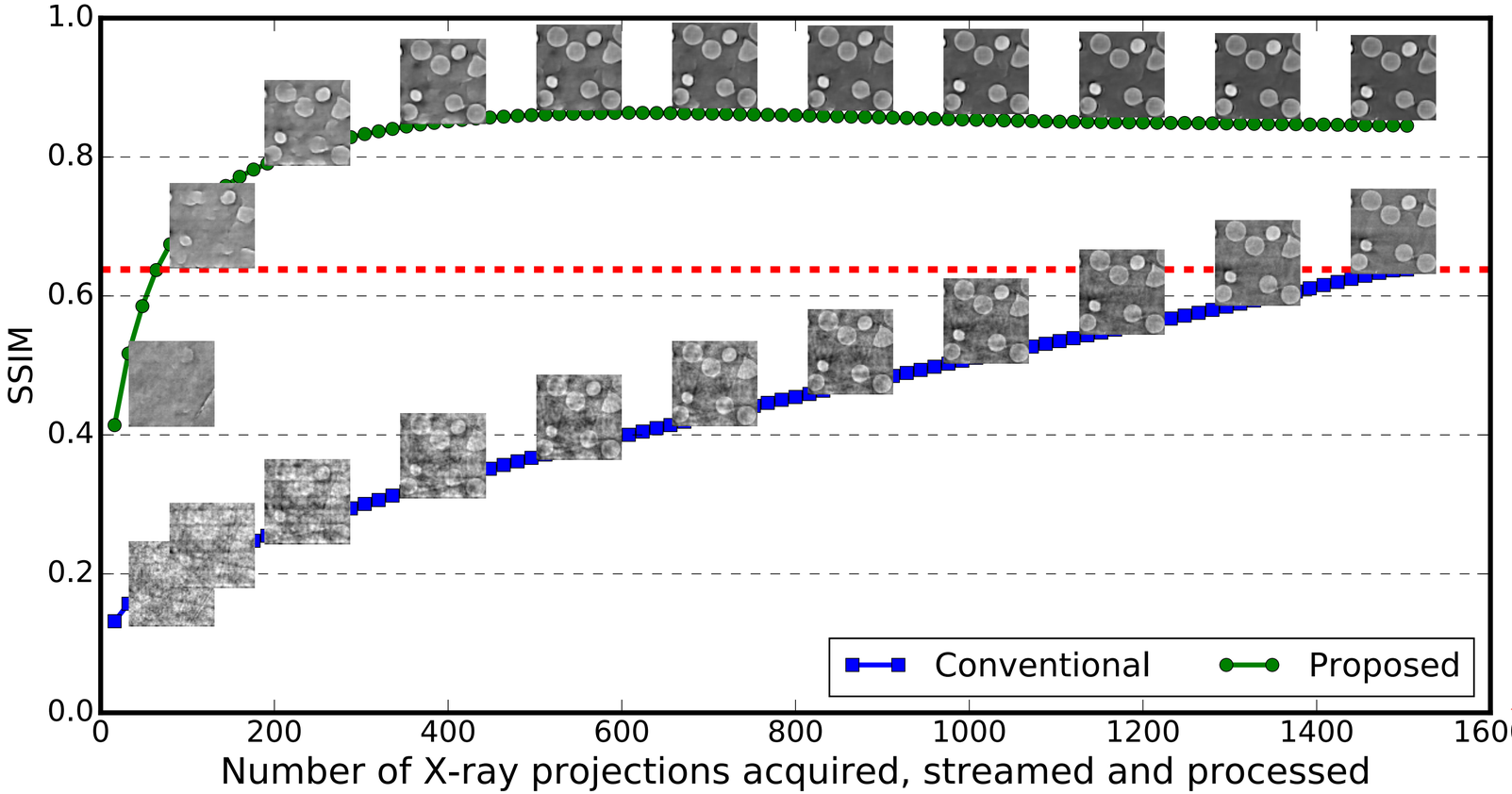


with data up to 1433s (1504 projections), before enhancement.

Three times faster turnaround time for domain scientists. A.K.A., three times increased throughput for the light source and computing facility.

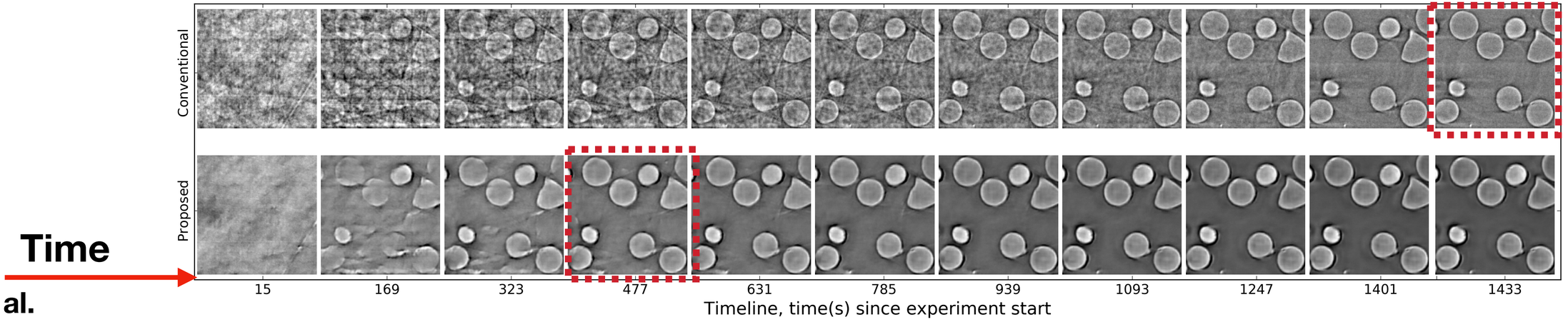
Important as enablers of experiment steering, where quick turnaround is required.

With **Tekin Bicer et al.**



- Best possible as ground truth to train TomoGAN
- With TomoGAN, reconstruction converts faster
- TomoGAN captured features, thus it works for any level of noisy although denoised image quality varies.

Time



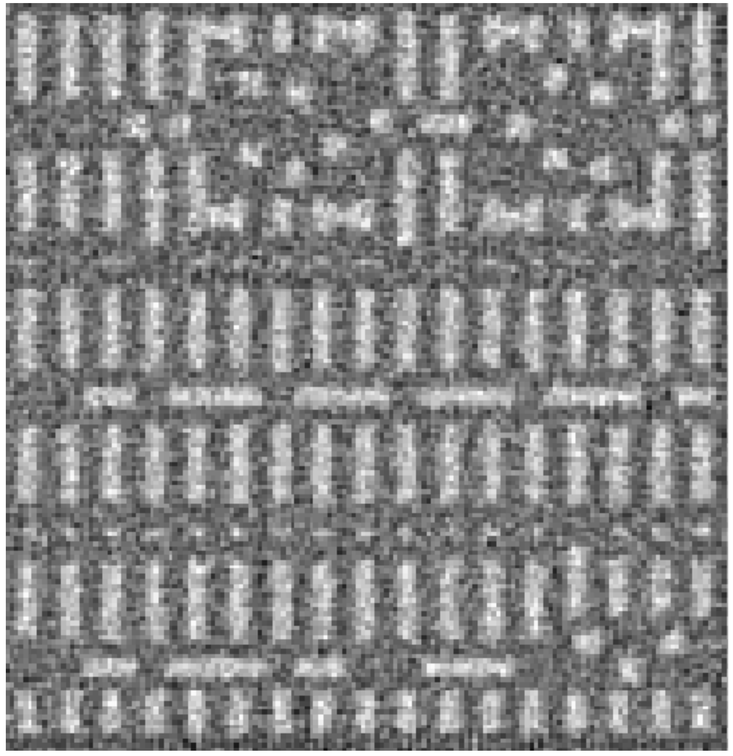
Time

With **Tekin Bicer et al.**

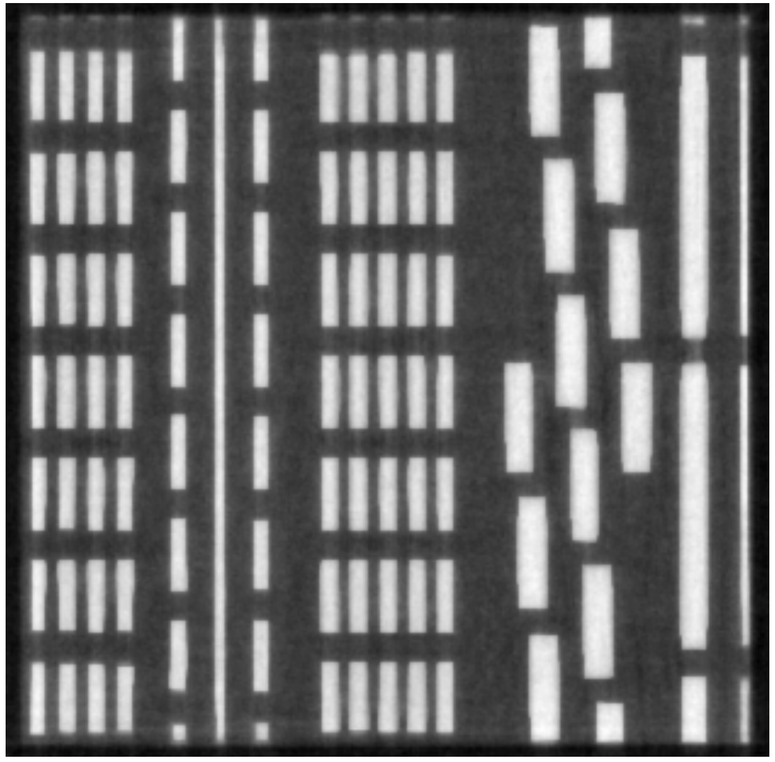
TomoGAN - Extended use case **2** - Prior in ADMM

Joint ptycho-tomography is a powerful framework to recover the refractive properties of the 3D object while relaxing existing requirements for lateral probe overlap.

- There is a ptychography process to reconstruct projections needed for tomography.
- Less datapoint results in noisier ptychography reconstruction and worse tomography images.
- Alternating direction method of multipliers (ADMM) is used as a generic reconstruction framework to efficiently solve the joint ptycho-tomography problem.
- TomoGAN here was used as a learnt prior in the ADMM loop to iteratively enhance tomography images.



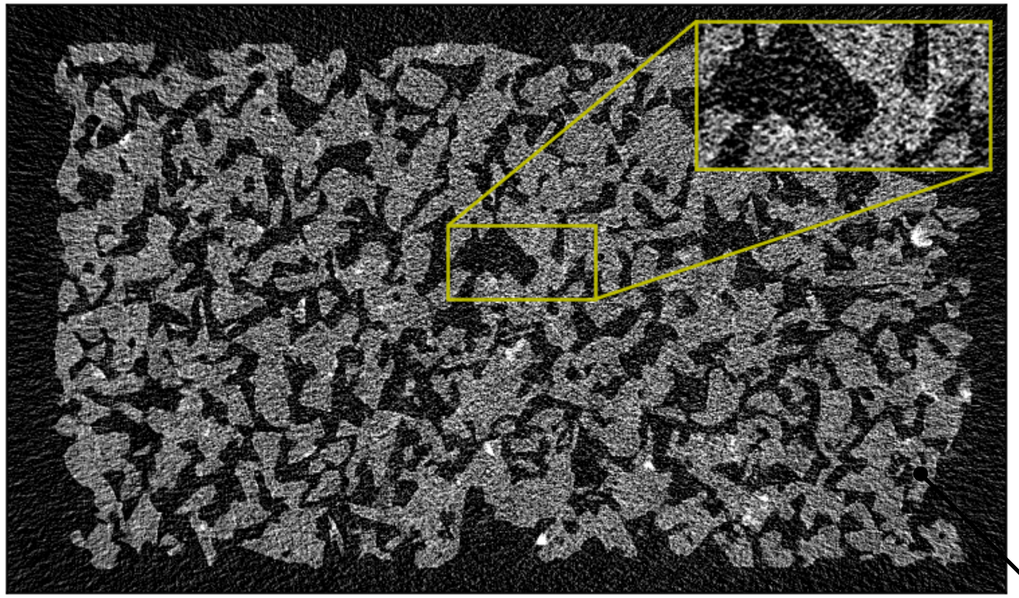
Original ADMM w/o using TomoGAN
One of images inTraining Dataset



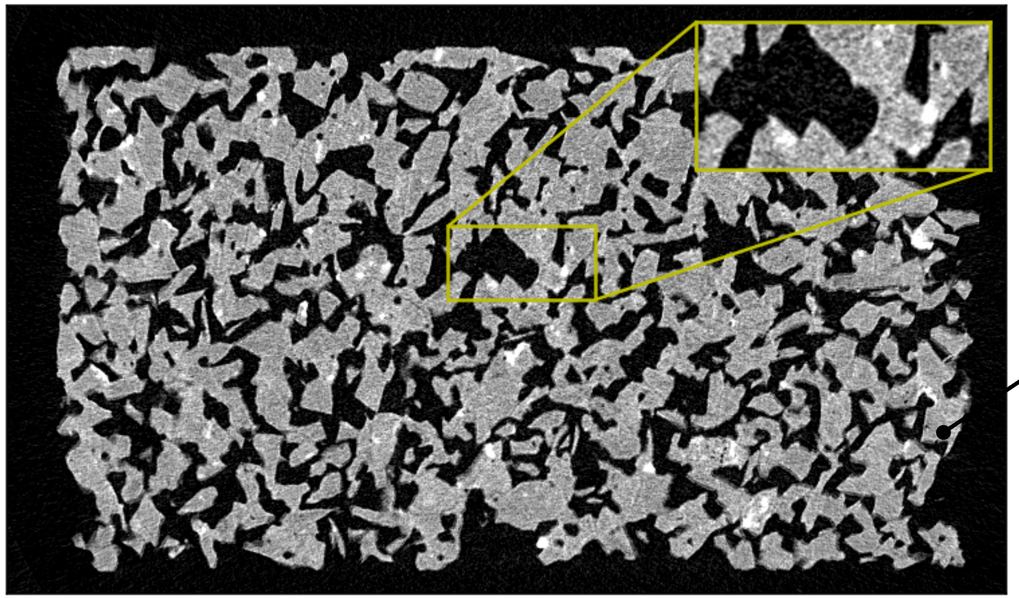
TomoGAN as learned prior in ADMM
One of images in Testing dataset

With **Selin Aslan et al.**

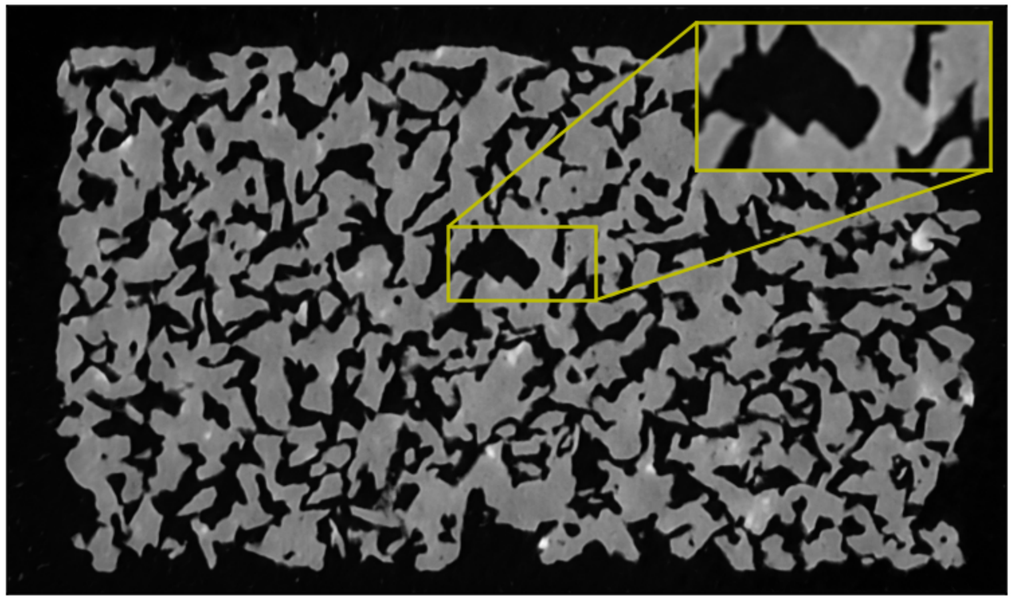
TomoGAN - Extended use case **3** - 3M with alignment issue



180°, large step size, no frame avg. (45 minutes)



Best attempt (4 hours)



TomoGAN enhanced (Model Output)

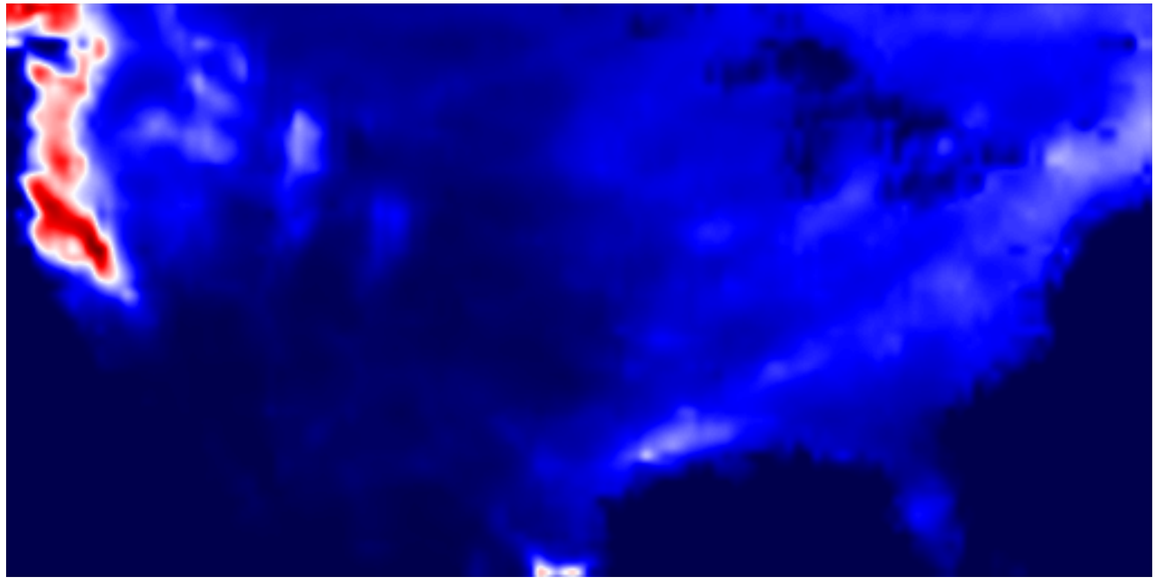
- (X,y) pair comes from two experiments;
- Impossible to perfectly aligned, like rotated a bit;
- Not a big problem for scientists but a big problem to ℓ_{mse}
- Tune the weight of ℓ_{mse}, ℓ_{vgg} and ℓ_{adv} works.

With **Myles Brostrom et al.**

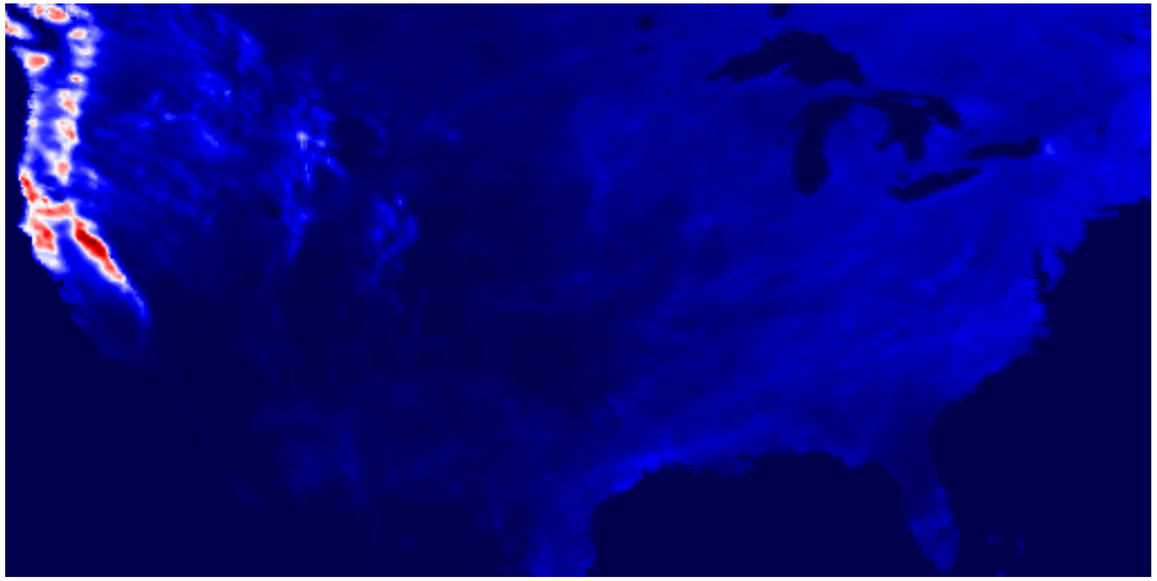
Conditional Super Resolution for Weather Research and Forecasting(WRF) Downscaling

Motivation

- WRF model can simulate much more accurately than coarse resolution model
- High resolution simulation results provide important input for other research fields, such as hydrological/urban modeling and risk assessment for critical infrastructures.
- However, simulate at high resolution is computation expensive, e.g., grid spacing of 50 km take 4,000 CPU hours, 12km takes 24,000 CPU hours, while 4km take 2,000,000 CPU hours.
- So, can Deep Learning help on downscaling? i.e., simulate at coarse resolution and enhance it afterwards.

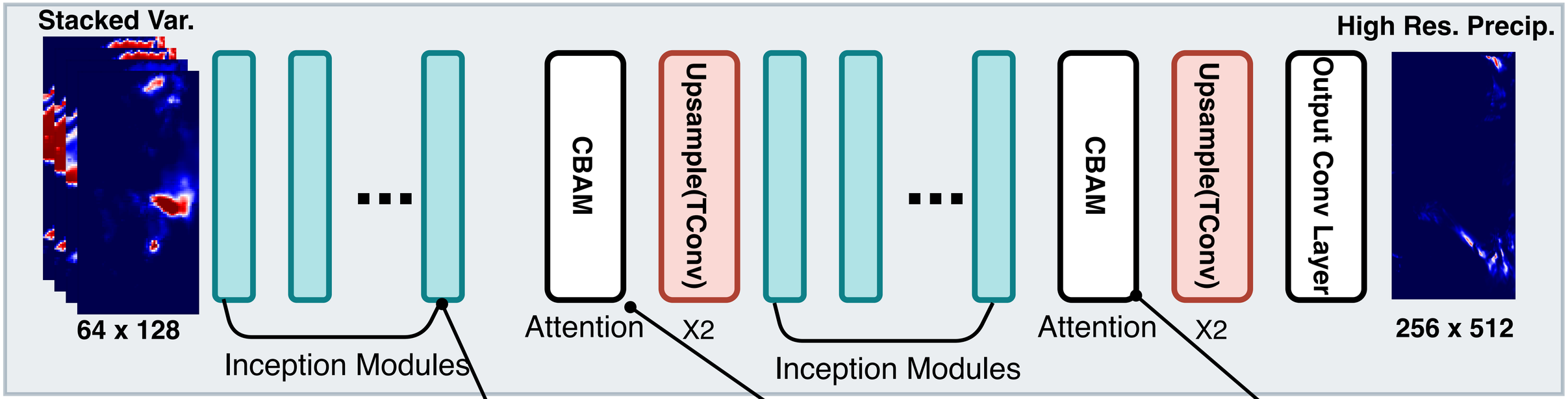


Simulation at 50 KM

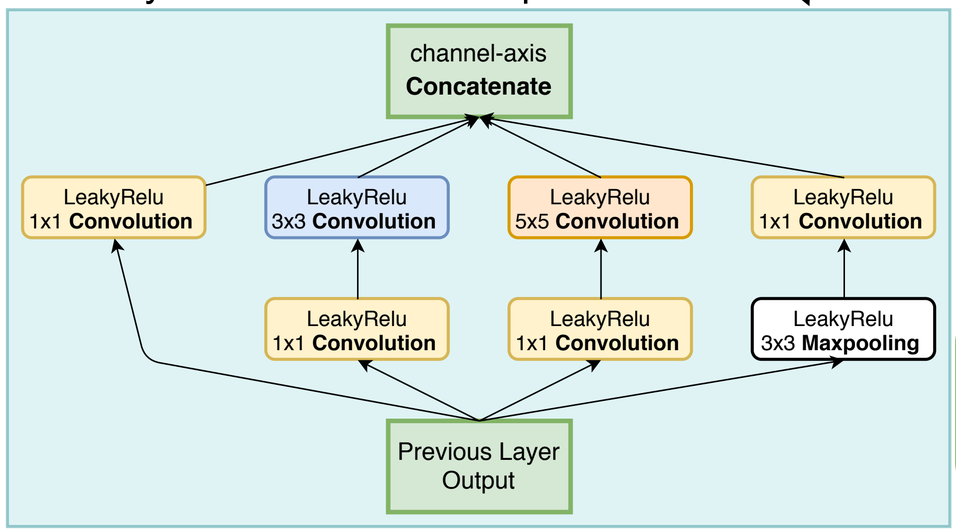


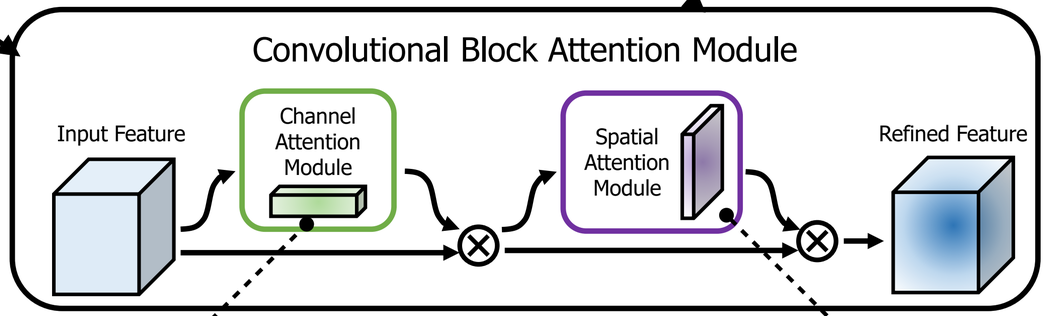
Simulation at 12 KM

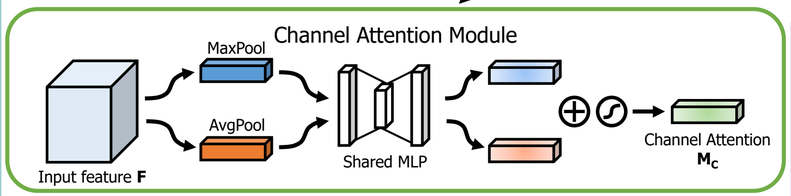
Conditional Super Resolution for WRF downscaling

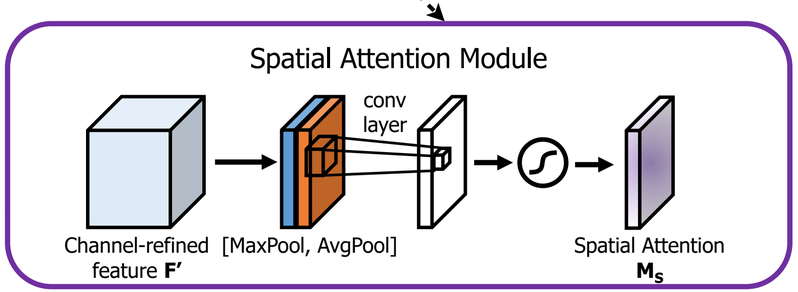


Each layer has different receptive field size



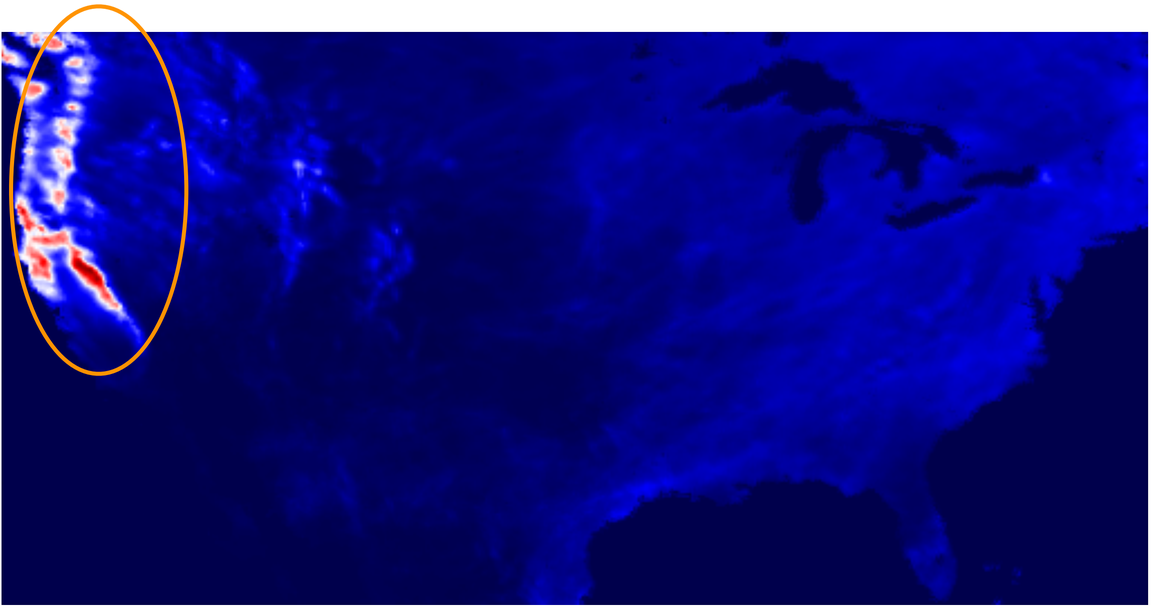






S. Woo et al. CBAM: Convolutional Block Attention Module ECCV 2018

Conditional Super Resolution for WRF downscaling



12 KM

Data:

(12KM, 50KM) pair, 3-hourly precipitation snapshot in one year

Jan. - Oct. for training <*seasonal difference*> Nov. and Dec. for testing

Training:

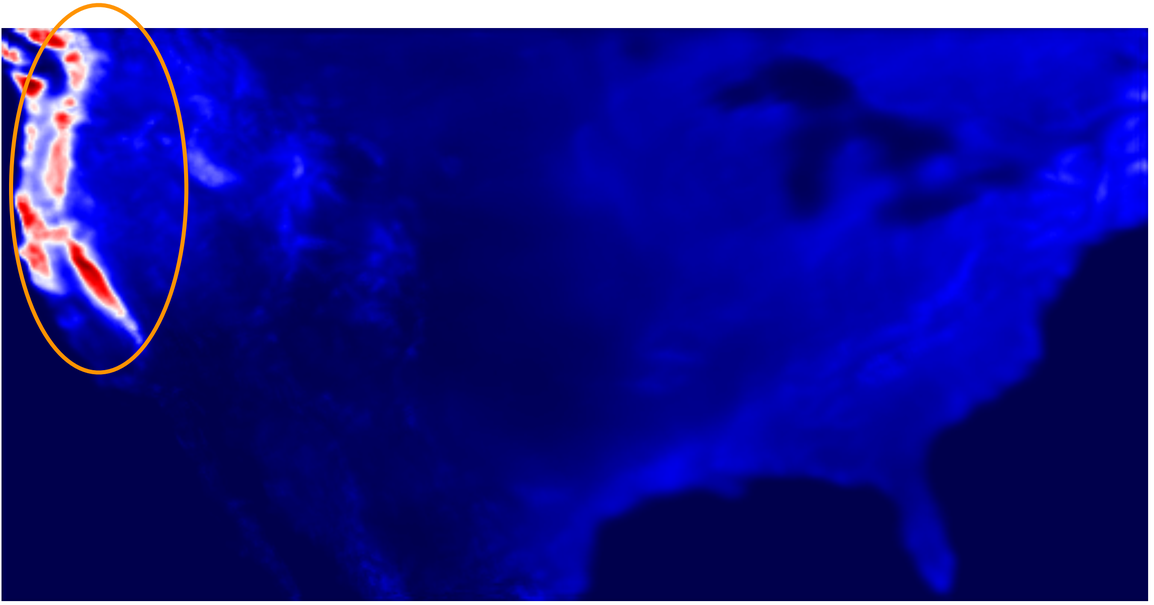
300 epochs

16 mini-batch size

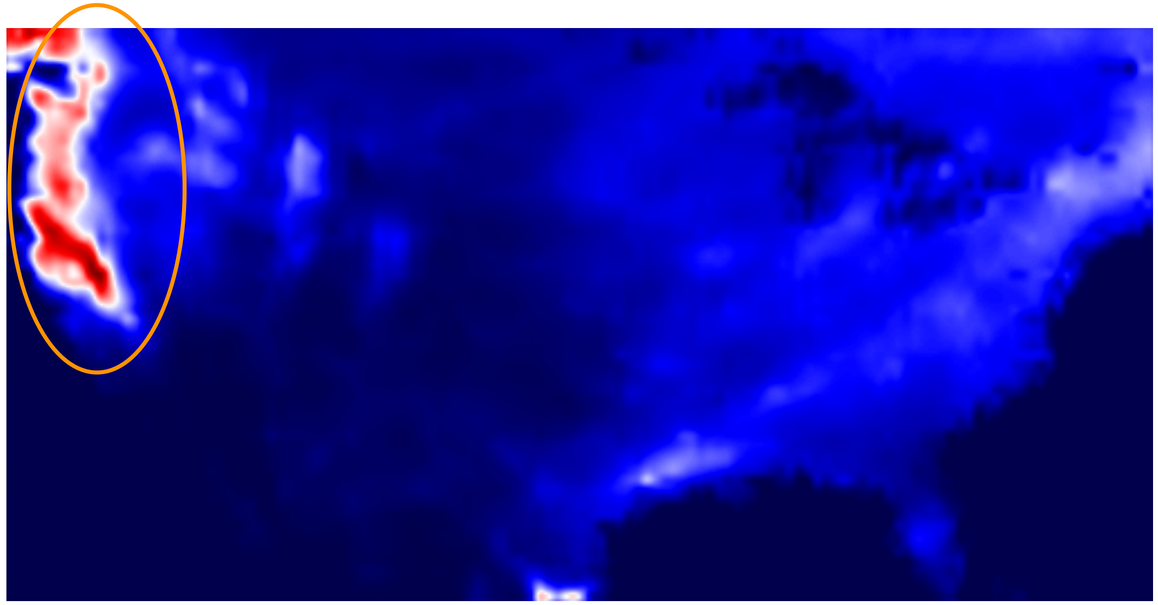
Testing:

Nov. and Dec.

SSIM: 0.86 vs. 0.79, MSE: 0.004 vs. 0.008, Pearson: 0.96 vs. 0.89



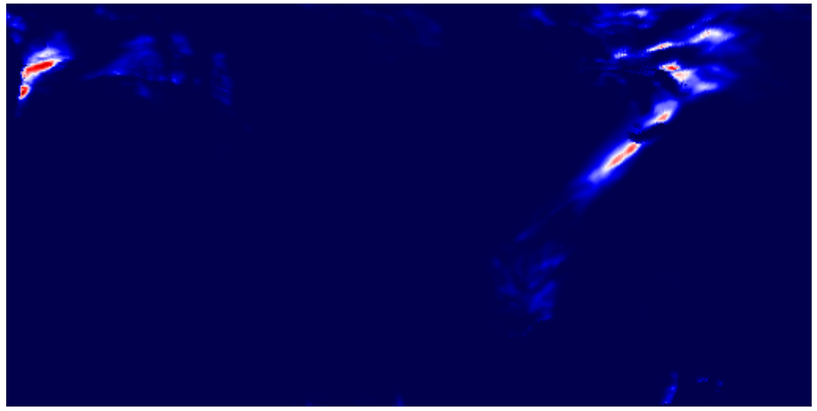
CSR from 50 KM



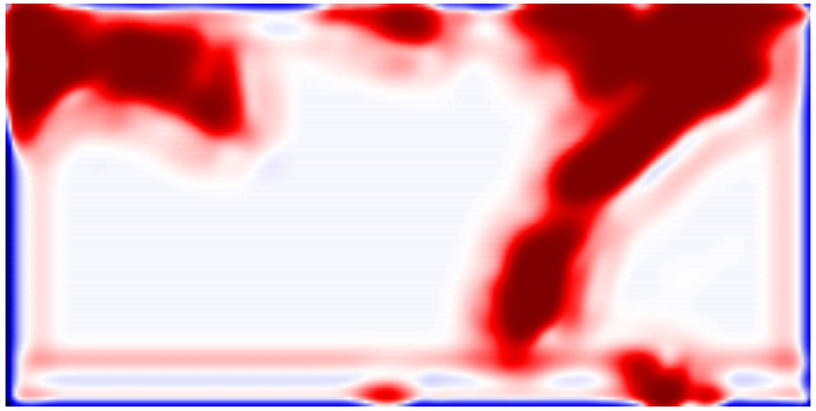
bicubic interpolation from 50 KM

Conditional Super Resolution for WRF downscaling

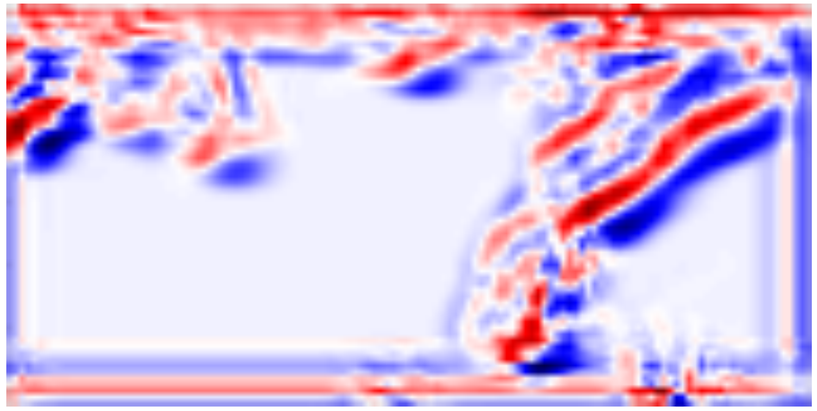
Learned spatial attention



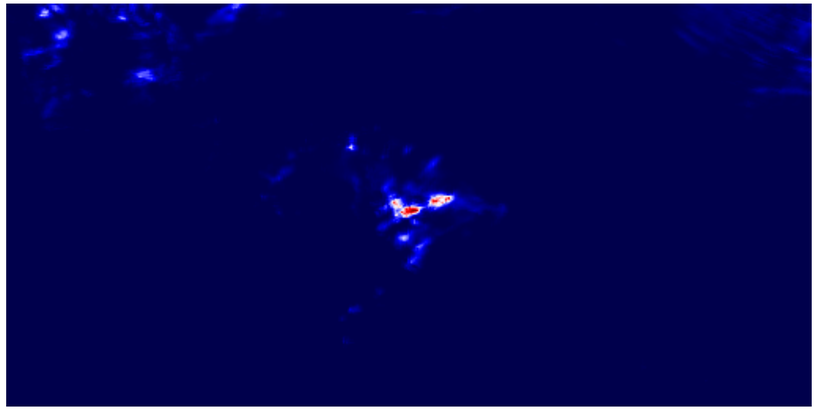
Expected Output (Ground Truth)



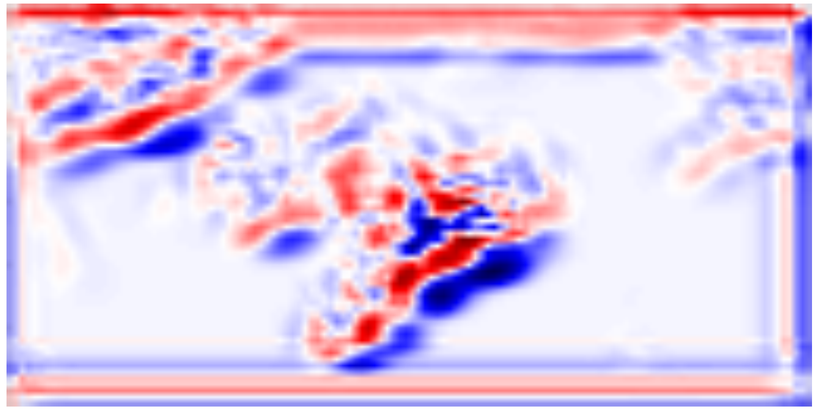
Spatial Attention 1



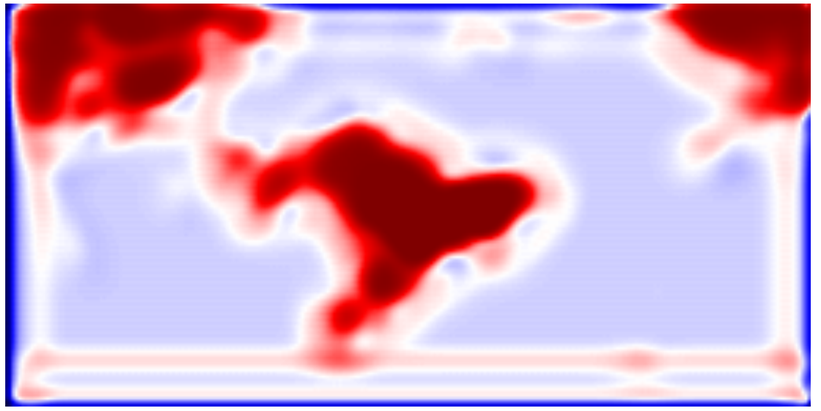
Spatial Attention 2



Expected Output (Ground Truth)



Spatial Attention 1



Spatial Attention 2

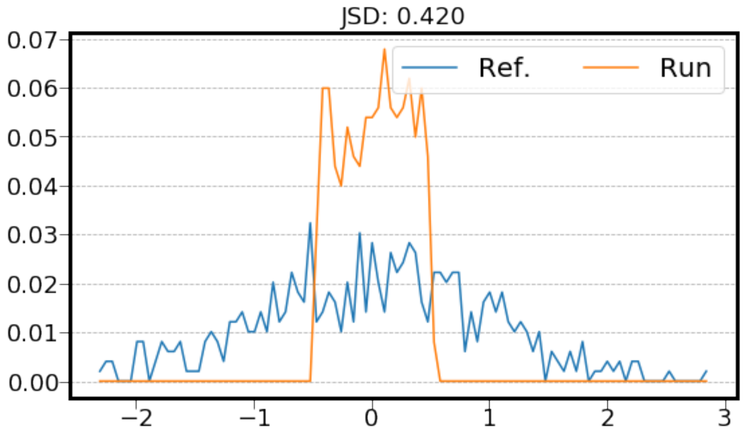
Self-driving Accelerator Operation

Powered by:

- [ ] 1,320 power supply controls the electron beam which provides X-ray radiation
- [ ] 20 years of monitoring every ~60s include: capacitor temperature, current, magnitude temperature, DAC, IGBT and voltage
- [ ] Currently dozens of failure annually
- [ ] It will be valuable for APS-U in its early stage or even testing stage.

Can we:

- [ ] Detect anomaly and raise alarms?
- [ ] Predict power supply failure before the weekly maintenance?
- [ ] Learn from expert, (adapt configuration) fix (some) potential power supply issue?



Plan and progress:

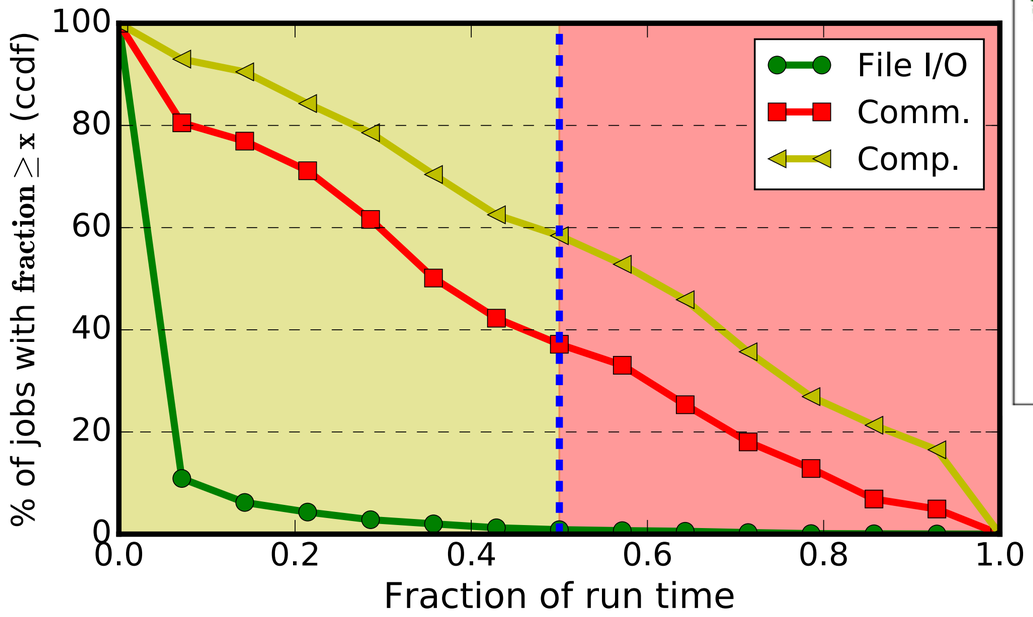
- [x] Auto-encoder for anomaly detection, to understand if recorded data can (fully) characterize power supply status
- [x] Conventional way for anomaly detection, statistical distance between known normal and realtime monitoring. Jensen–Shannon Divergence (JSD) works fine using 12 hours monitoring.
- [ ] Machine learning prediction for weekly maintenance intensive care.
- [ ] Learning from expert for auto-tuning

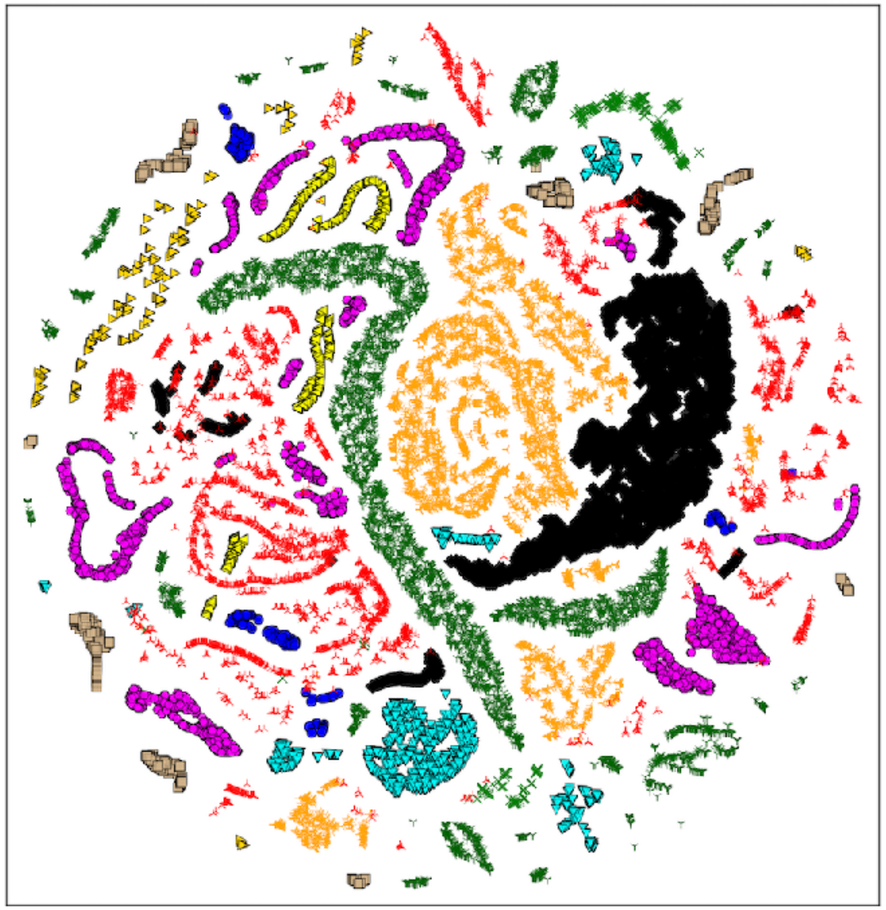
With **Michael Borland, Yipeng Sun et al.**

Automating HPC Taxonomy at LCFs for Fine Allocation

Motivation

- LCFs collected many logs/datasets for different purpose.
- Verify (fingerprinting) if the users are using the system in an appropriate way (actual application runs closely resembles the proposed usage).
- Categorize applications, for example as I/O intensive, communication intensive, computing intensive, or memory bound.
- Such categories can be used to target applications for optimization like *-aware schedule and allocation, power-cap setting, particularly if they are to be executed repeatedly (true for LCFs).
- We break down runtime to create labels (i.e., categorize), we then use hardware counters as feature to train ML classifier.
- We also propose fingerprint using information at the time of submission for querying and identification.





Z. Liu et al. Characterization and Identification of HPC Applications at Leadership Computing Facility. ICS'2020

Thanks!