

CAAP Quarterly Report

Date of Report: *Jan 7, 2021*

Prepared for: *Thomas Finch (Project Manager) and Joshua Arnold (CAAP Program Manager), U.S. DOT Pipeline and Hazardous Materials Safety Administration*

Contract Number: *693JK31850005CAAP*

Project Title: *Low-variance Deep Graph Learning for Predictive Pipeline Assessment with Interacting Threats*

Prepared by: *Hao Zhang (Colorado School of Mines) and Yiming Deng (Michigan State University)*

Contact Information: *Dr. Hao Zhang, Department of Computer Science, Colorado School of Mines; 1500 Illinois St., Golden, CO 80401; Phone: 303-273-3581; Email: h Zhang@mines.edu*

For quarterly period ending: *Jan 7, 2021*

Business and Activity Section

(a) Contract Activity

No contract modification was made or proposed in this quarterly period. No materials were purchased during this quarterly period.

(b) Status Update of Past Quarter Activities

In this first quarter of the one-year extension period, we focus on Subtask 3.2 to use machine learning algorithms to generate annotated simulation data with reliability/uncertainty analysis.

(c) Cost Share Activity

PI Zhang used his effort as the 20% in-kind cost share to work on the project at the Colorado School of Mines. Co-PI Yiming Deng used effort as the 20% in-kind cost share to work on the project at the Michigan State University. The cost share was used following the approved proposal and no modification was made.

(d) Performed Research

Fatigue Propagation Data Management

The quarter mainly focuses on different defect cases and their propagation study to be incorporated to deep learning algorithms. At first the report focuses on the propagation of regular

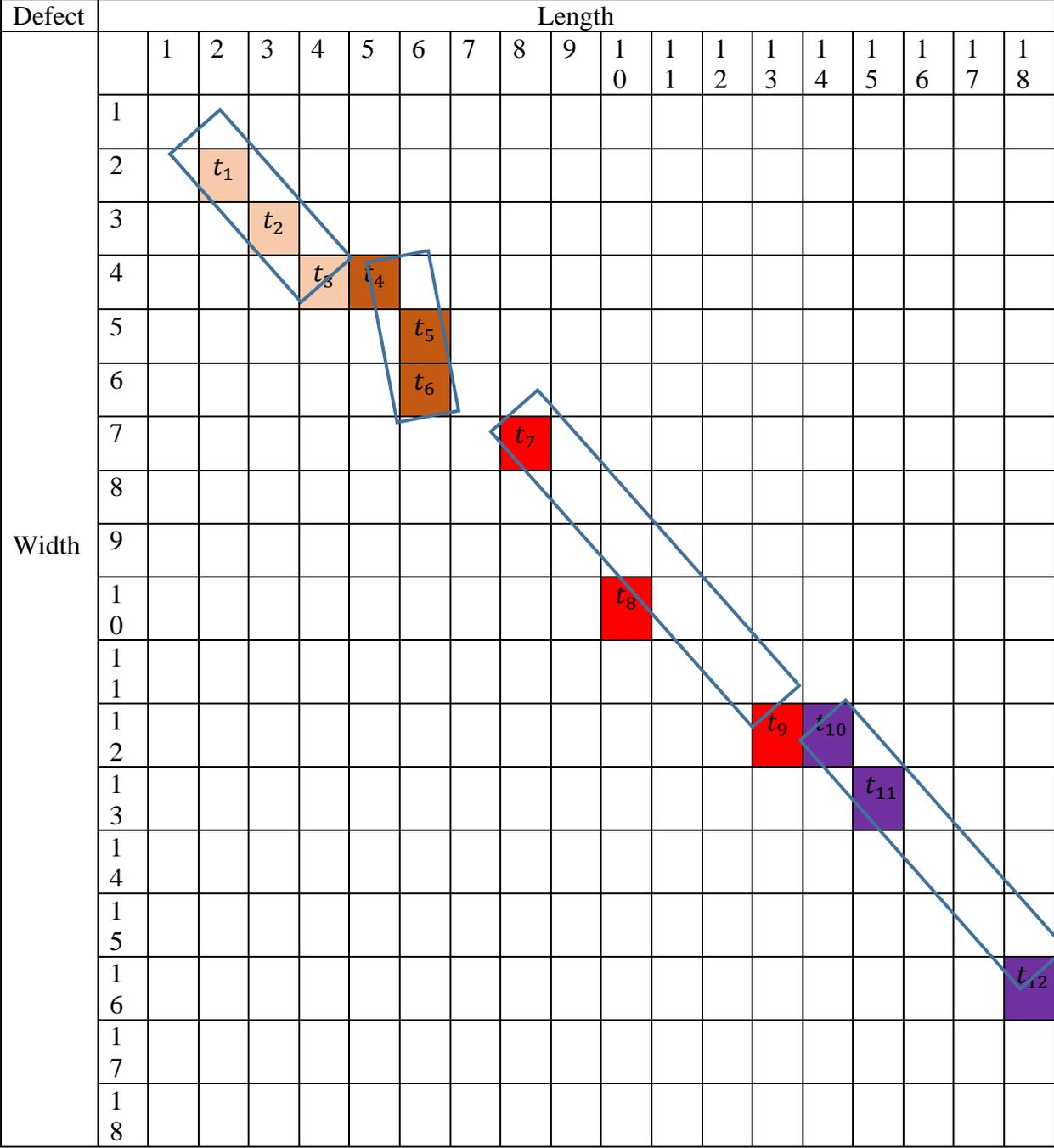
threats. Then the distortion of the magnetic flux due to the presence of defects in all the three directions B_x, B_y & B_z are captured and then are segmented into various classes by segmentation algorithms to be feed into GAN (Generative Adversarial Network) to create fake data as that of COMSOL generated data. Then the report shows the simulation of Surface Crack Corrosion (SCC) in COMSOL and the FEM mesh generations on stainless-steel sample. Finally, the efficacy of GAN in producing fake data as similar to as that of real data taking into account various hyperparameters and resolution of the images is portrayed.

Propagation of regular threats

The different complex defect propagation parameters are shown in Table 1. Here the defects are grown in a complex manner with defect center varying in each case which makes the design more complex to predict the defect growth by deep learning algorithms such as RNN.

Table 1: Different propagation parameters of the growing defects

Defect	Position x												
		-5	-4	-3	-2	-1	0	1	2	3	4	5	
Position y	-5												
	-4												
	-3												
	-2												
	-1												
	0												
	1												
	2												
	3												
	4												
	5												



Defect	Position x												
		-5	-4	-3	-2	-1	0	1	2	3	4	5	
Position y	-5												
	-4												
	-3												
	-2												
	-1												
	0												
	1												
	2												
	3												
	4												
	5												

ID	Defect Center (mm)	Defect parameter (mm)	Defect Shape
1	(0,0)	2 * 2	Rectangular
2	(0,0)	3 * 3	Rectangular
3	(0,1)	4 * 4	Rectangular
4	(1,1)	4 * 5	Rectangular
5	(1,1.5)	5 * 5.5	Rectangular
6	(1,2)	6 * 6	Rectangular
7	(1.5, 3)	7 * 8	Rectangular
8	(2,4)	10 * 10	Rectangular
9	(2, 4.5)	11.5 * 13	Rectangular
10	(3, 4.5)	12 * 14	Rectangular
11	(3.5, 5)	13 * 15	Rectangular
12	(3.5, 5)	16 * 18	Rectangular

Based on the above table different figures with the growing defects and shifted centers are shown below. For defects with ID 1 – 3 defect parameters are modelled based on first three rows of the table. ID 4 – 6 follows the propagation based on next three rows of the table and so on. All these designs are made in such a way so as to make the growth propagation hard to predict.

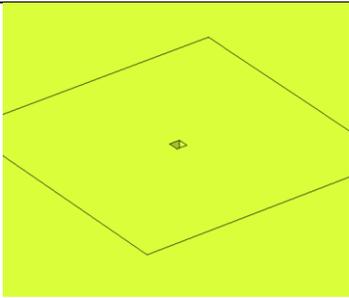
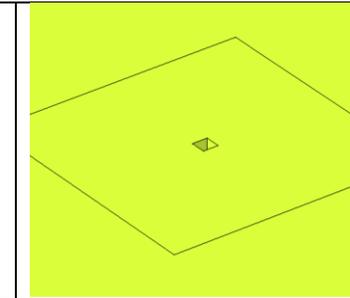
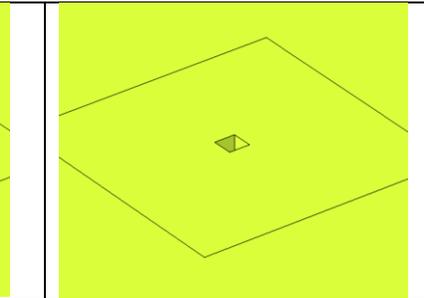
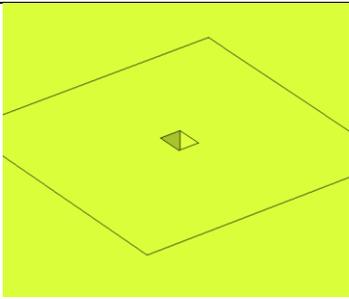
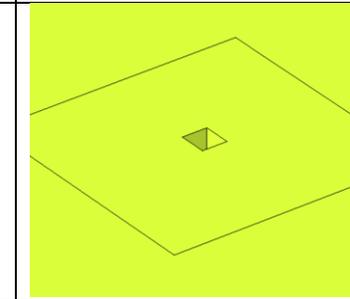
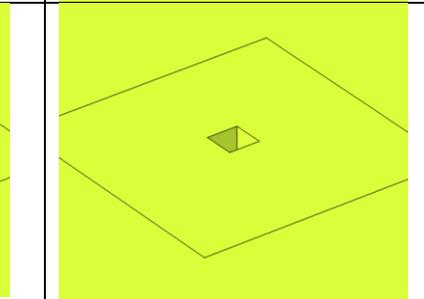
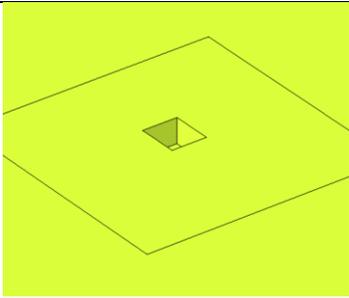
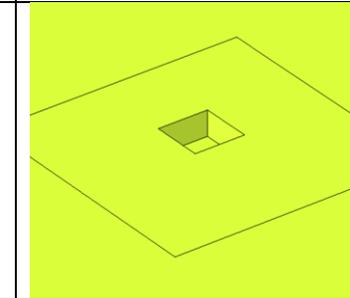
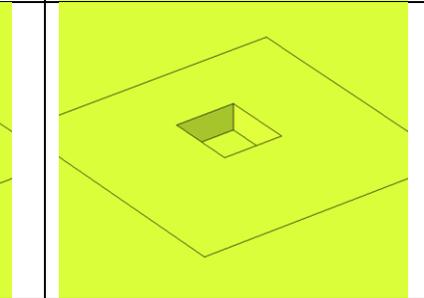
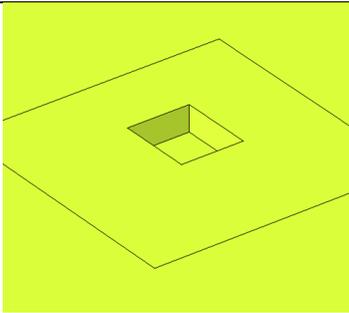
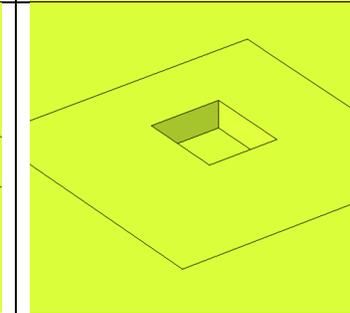
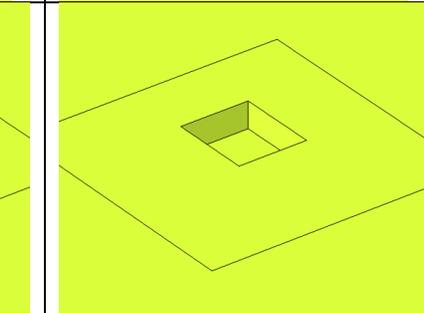
ID			
1-3			
4-6			
7-9			
10-12			

Figure Error! No text of specified style in document. : Different propagation growth patterns for regular defect constructed in COMSOL

Data driven prognostics & diagnostic solutions in this project are likely to underperform due to limited data availability while in the real-life scenario, the number of failure data samples is insufficient as well. To address this problem, we adopt GAN for generating real-valued data which allows training datasets to be augmented so that the number of data samples is increased. In contrast to existing data generation techniques which duplicate or randomly generate data, GAN can generate new and realistic failure data samples. In the end, we still need to optimize

the GAN which utilize the auxiliary information pertaining to the different stages of defect propagation.

Application of GAN on MFL data

GAN was introduced in one of the earlier quarters. Here we will focus on the application of GAN on our MFL data. Our goal is to produce fake data out of the generator (G) by fooling the discriminator (D) as much as we can. However, the fake data has to be very close to the real data so that those can be applied to data hungry deep learning algorithms.

The idea behind GANs is that two networks, a generator G and a discriminator D is competing against each other[4]. The generator makes fake data to pass to the discriminator. The discriminator also sees real data and predicts if the data it's received is real or fake. The generator is trained to fool the discriminator, it wants to output data that looks as close as possible to real data. And the discriminator is trained to figure out which data is real and which is fake[1]. What ends up happening is that the generator learns to make data that is indistinguishable from real data to the discriminator.

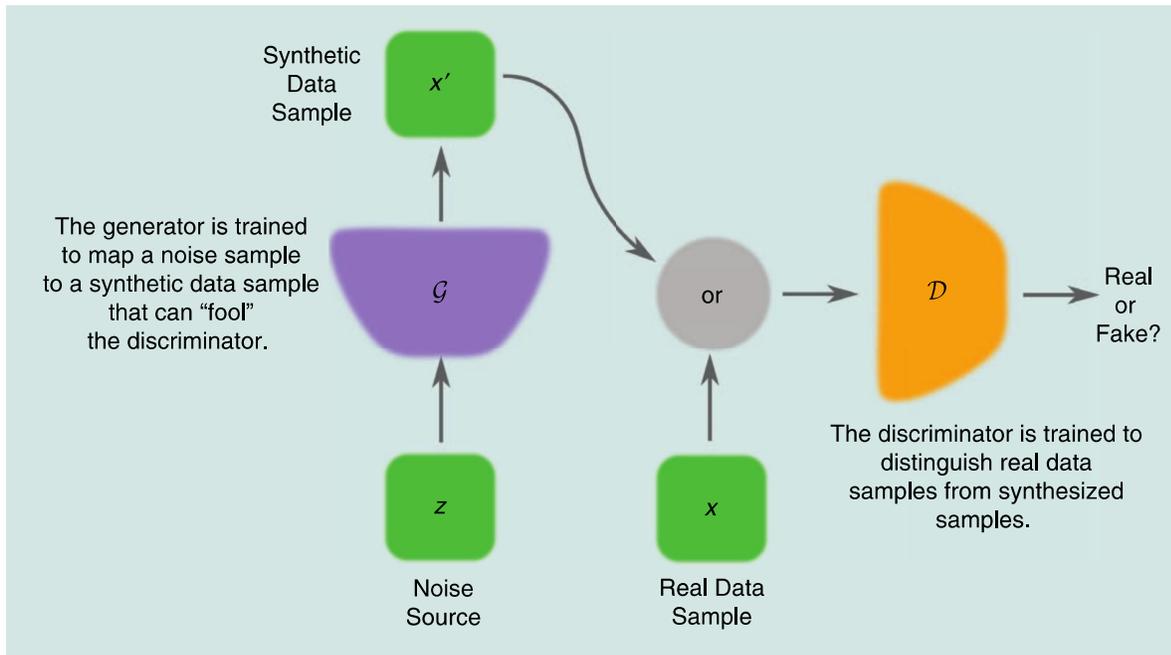
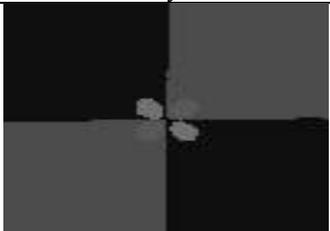
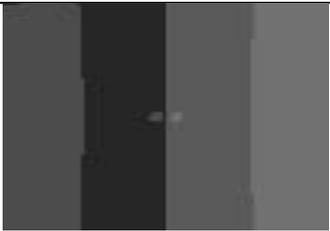
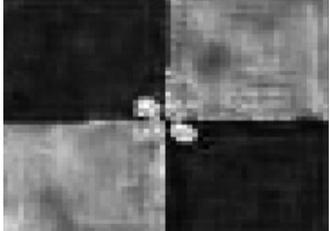
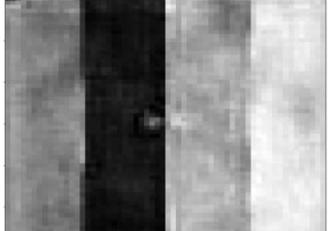
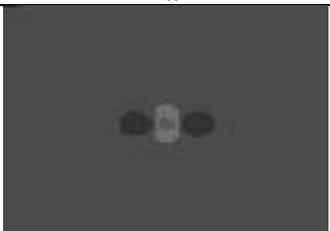
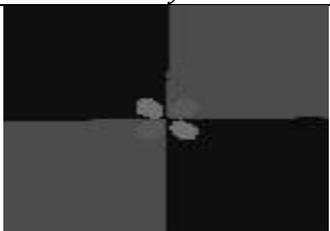
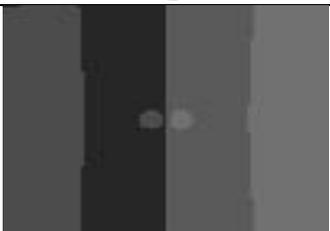
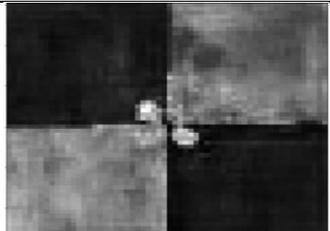
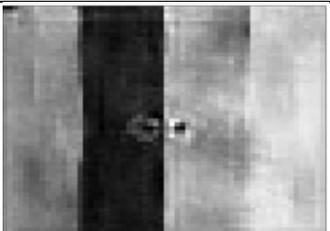


Figure Error! No text of specified style in document. Schematic of the working procedure of GAN[5]

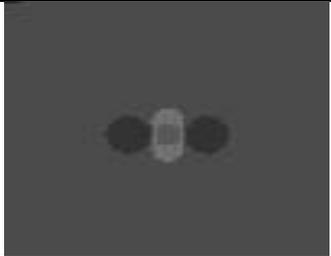
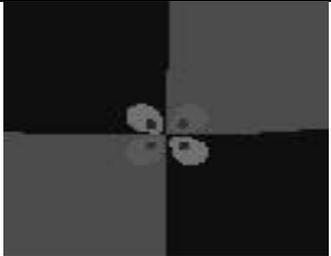
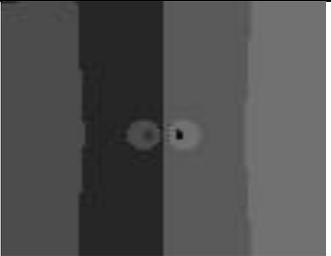
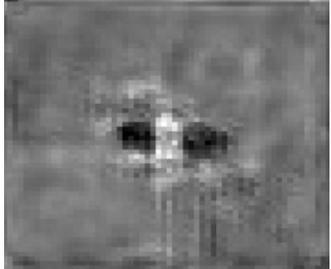
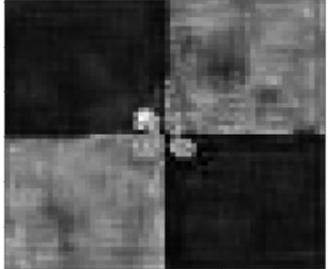
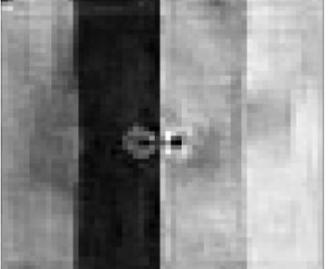
The COMSOL based NDE framework involves the FEM study and hence for defect propagation there is mesh and time consumption trade-off. To track very small defects the mesh around the defect has to be extremely fine (extra fine) and this leads to more time consumption in simulation framework. Hence our objective is to develop a data augmentation framework using GAN so that we can develop as much fake data we can by taking into account lesser number of real synthetic data [2]. As shown in the figure, generative models learn to capture the statistical distribution of

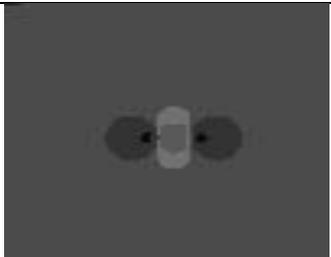
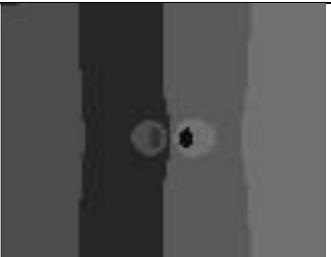
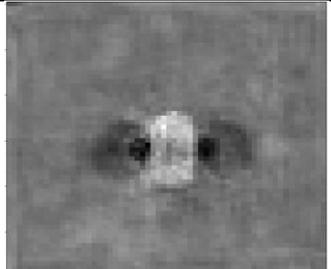
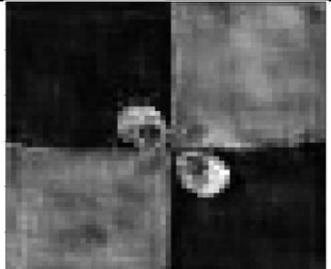
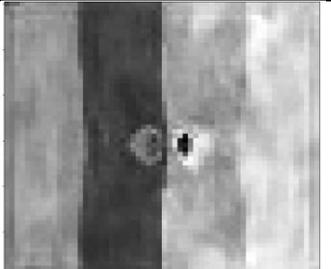
training data, allowing us to synthesize samples from the learned distribution. On top of synthesizing new measurements (when experimental data is finalized), we are also interested in using the representations that such models learn for tasks such as classification. The discriminator network comprises of 4 conv2D layers each of which is followed by leaky relu activation and a dropout layer. Then the final output of Conv2D layer is flattened and passed through a dense layer and final passed through a softmax classification.

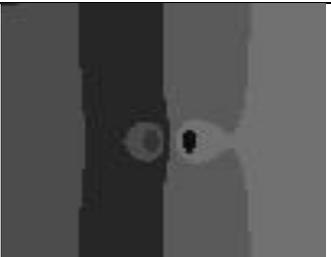
ID = 1			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

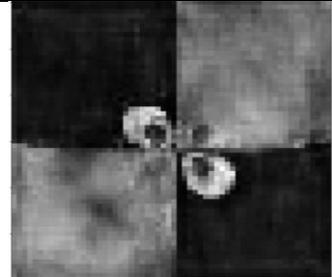
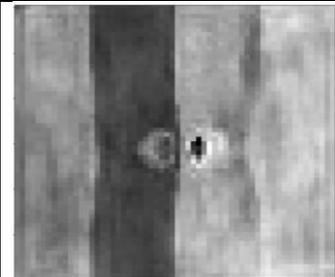
ID = 2			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

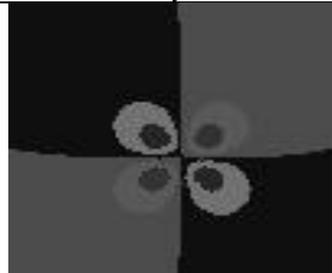
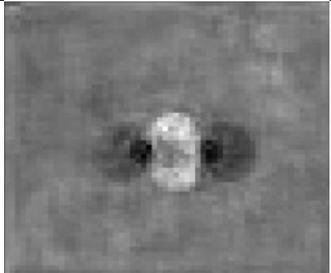
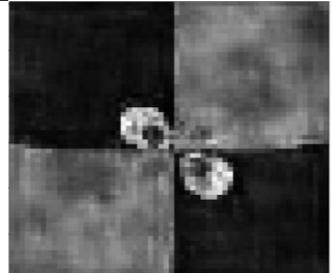
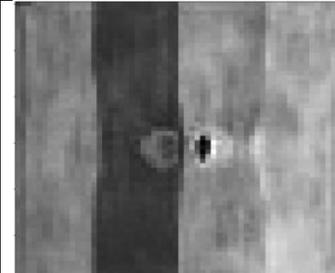
ID = 3			
	B_x	B_y	B_z

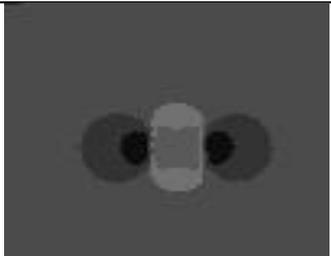
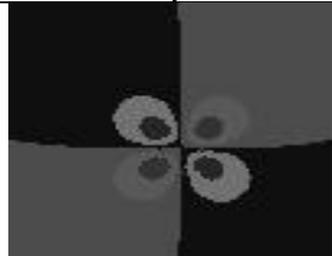
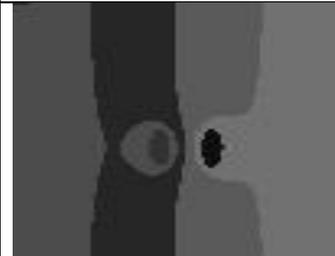
Real x			
Fake $G(z)$			

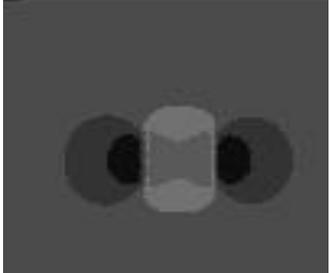
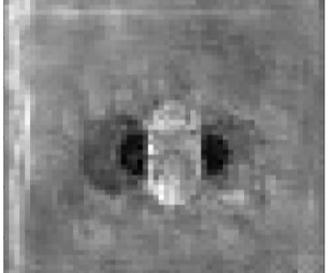
ID = 4			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

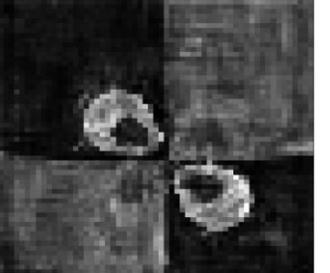
ID = 5			
	B_x	B_y	B_z
Real x			

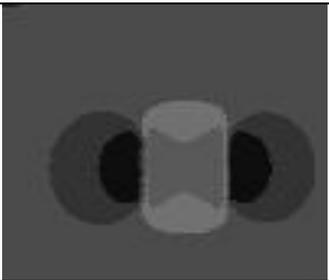
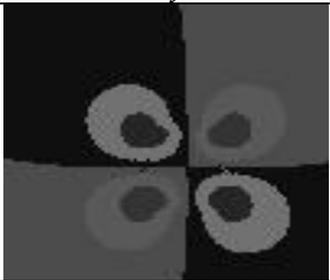
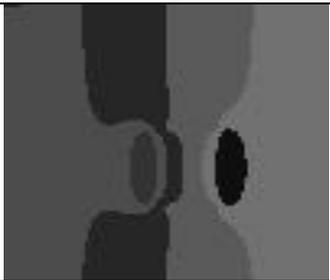
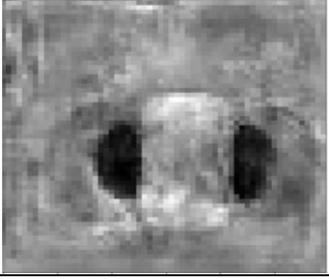
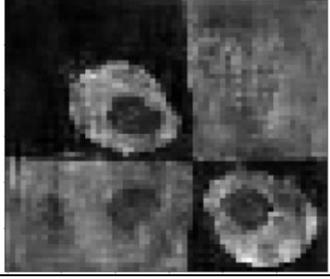
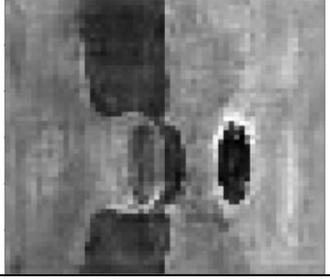
Fake $G(z)$			
-------------	---	--	---

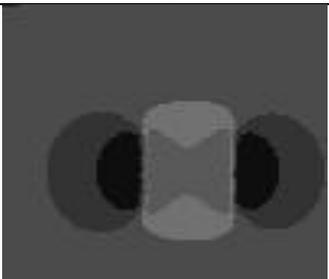
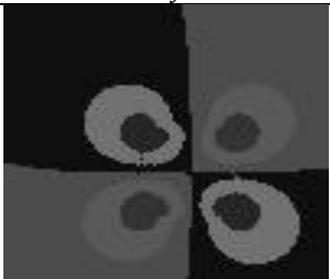
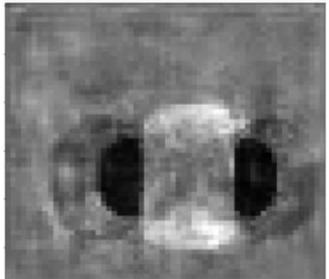
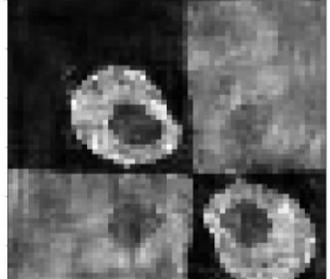
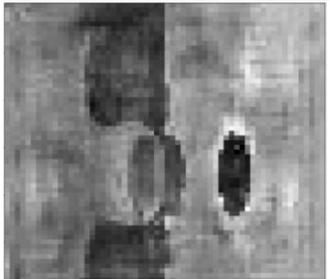
ID = 6			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

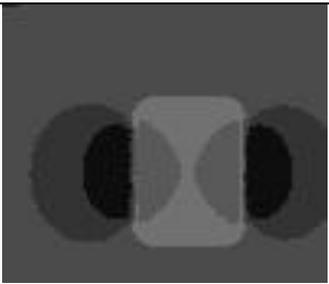
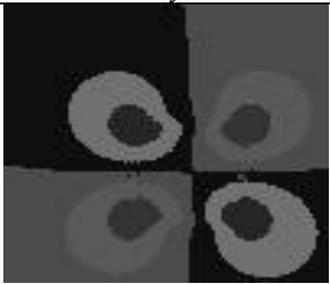
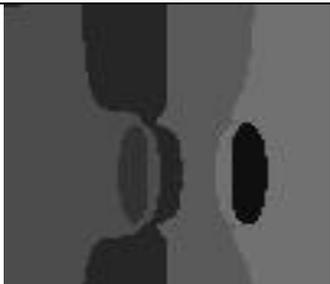
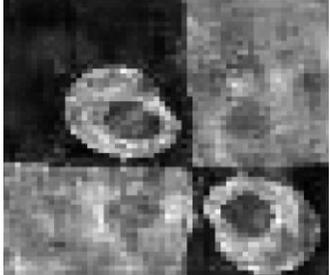
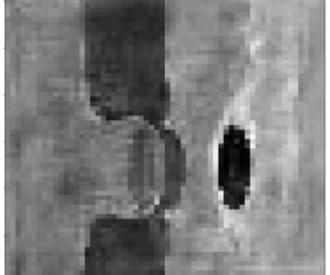
ID = 7			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

ID = 8			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

ID = 9			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

ID = 10			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

ID = 11			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

ID = 12			
	B_x	B_y	B_z
Real x			
Fake $G(z)$			

Discussion on GAN

GANs are an emerging technique for both semi-supervised and unsupervised learning. They achieve this through implicitly modeling high-dimensional distributions of data. To optimize GAN in our project, some metrics should be considered which reflect the diversity as well as quality of the outcome fake images. For any metric, a model's generalization gap is the difference between the metric's value on the true data distribution less its value on the training set. As a model changes, classical machine learning theory divides the behavior of the generalization gap into two regimes: underfitting and overfitting. Overfitting is when the metric is improved on the training set at the expense of its value on the true data distribution. Moreover, in the previous literature[6], the performance GAN with different amounts of data being investigated regarding 0% to 100% of training data, the gap between the discriminator's training and validation accuracy keeps increasing, suggesting that the discriminator is simply memorizing the training images. This happens not only on limited data but also on the large-scale dataset. In future study, we should extend GAN framework to the conditional setting by making both the generator and the discriminator networks class or label conditional. Conditional GANs have the advantage of being able to provide better representations for multimodal data generation. The additional information in this project can be interpreted as latent code. This latent code can be used to discover object classes in a purely unsupervised fashion, although it is not strictly necessary that the latent code be categorical.

Reference

- [1] "The "Risk Modeling Work Group" Discussion of Interactive Threats," 2013.
- [2] Y. Chen, H. Zhang, J. Zhang, X. Liu, X. Li, and J. Zhou, "Failure assessment of X80 pipeline with interacting corrosion defects," *Engineering Failure Analysis*, vol. 47, pp. 67-76, 2015, doi: 10.1016/j.engfailanal.2014.09.013.

- [3] I. N. G. A. o. America, "Interacting Threats to Pipeline Integrity – Defined and Explained."
- [4] J. P.-A. Ian J. Goodfellow, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair†, Aaron Courville, Yoshua Bengio‡, "Generative Adversarial Nets," 2014.
- [5] S. Z. Z. L. J. L. J.-Y. Z. S. Han, "Differentiable Augmentation for Data-Efficient GAN Training," *34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.*, 2020.
- [6] G. a. M. a. H. Y. a. C. Meinel3, "Dropout-GAN: Learning from a Dynamic Ensemble of Discriminators," 2020.