Machine learning for physics simulation anomaly detection

Adam Good | Howard Pritchard | Lissa Moore | Garrett Kenyon

Los Alamos National Laboratory, Los Alamos NM, USA

LA-UR-21-27175

Outline

- Introduction
  - Hydrodynamics
  - Direct Numerical Simulation
  - High Performance Computing
  - Anomaly Detection
- Methodology / Experiments
  - Dataset
  - Methods
  - Models
- Conclusions
  - Results / Analysis
  - Future Work

Los Alamos
NATIONAL LABORATORY
EST. 1943

Introduction

LA-UR-21-27175

Hydrodynamics

- Hydrodynamics is the subsection of Fluid Dynamics that studies the flow of liquids
- Used in many areas of study and application
  - Ocean Currents
  - Blood Flow
  - Rocket Engines
  - More
- These problems often not have known analytical solutions…
- Example: Navier-Stokes Equations

Direct Numerical Simulation (DNS)

- DNS is used to find solutions to hydrodynamic (and other) problems with no known analytical solutions
- Uses numerical methods to compute accurate approximations of the solutions
- Often is computationally intensive and requires extensive computational resources
  - Requires High Performance Computing (HPC)

HPC, Faults, and Anomalies

- HPC Clusters tend to have large amounts of computational resources such as CPUs, Memory, and more
- There is a significant probability of memory faults at the high scales found in HPC
- Some faults could simply cause job failure, but others may cause silent data corruption (SDC) anomalies
  - Costly
- It's also possible for anomalies to come from other sources

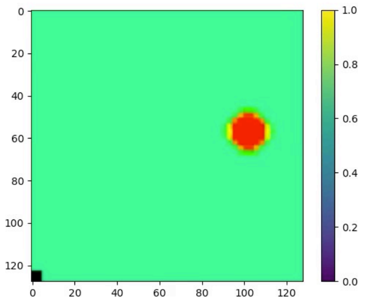
Anomaly Detection

- Computer-Vision techniques have been used to detect anomalies in images
  - Concrete Deficiencies
  - Skin Diseases
  - More
- One approach is reconstruction loss
  - Train an autoencoder to recreate nominal images
  - If the autoencoder is unable to recreate an image, it is likely anomalous
- We applied this idea to hydrodynamic simulations

Methodology

Dataset

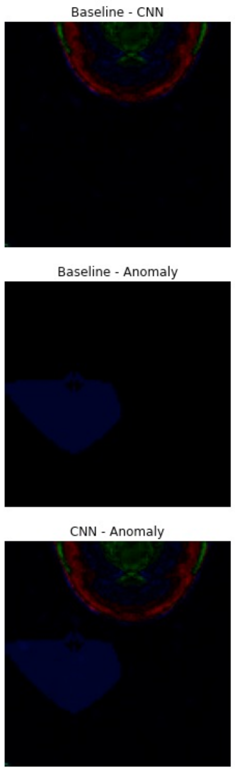
- We created a dataset of simulations using a tool called CLAMR
  - Simulations of the shallow water equation
- Consists of 459 pairs of simulations
  - Nominal and anomalous
  - 100 frames each
  - Varies on mesh size, domain size, and initial state
- Each of the 100 frames was turned into an image
  - 128x128 pixels



One simulation from the dataset

Method

- Reconstruction Loss Anomaly Detection
  - Train a model to reconstruct nominal data
    - Usually using an autoencoder
  - If an input is anomalous, the autoencoder should fail to create an accurate reconstruction
  - Train a classifier to determine if the error between the input and reconstruction implies anomalous input



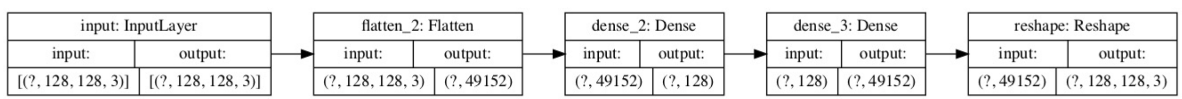
A comparison of differences
Baseline – CNN
Baseline – Anomaly
CNN – Anomaly

Methods (cont.)

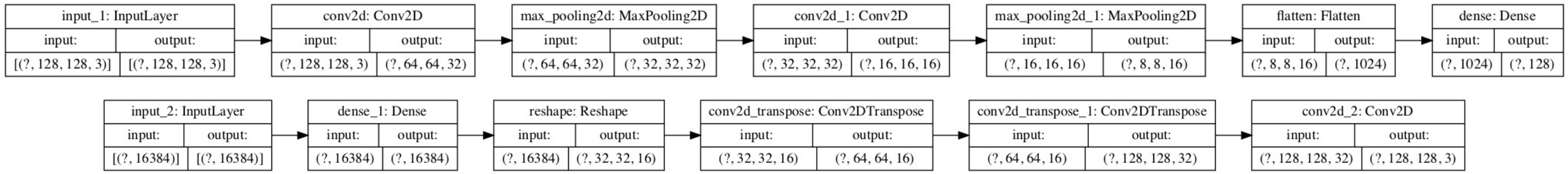
- Predictive Reconstruction
  - Since an anomaly may have the same ”shapes” as nominal data, we modified the technique
- Predict a subsequent frame based off the previous two frames
- Find the difference between the prediction and the same frame from the simulation
- Relies on model learning how to progress the simulation rather than learning nominal reconstructions

Autoencoders

- We tried two traditional autoencoders
  - Baseline feed-forward ANN
  - Convolutional
- Both predict a frame given the frame two timesteps back



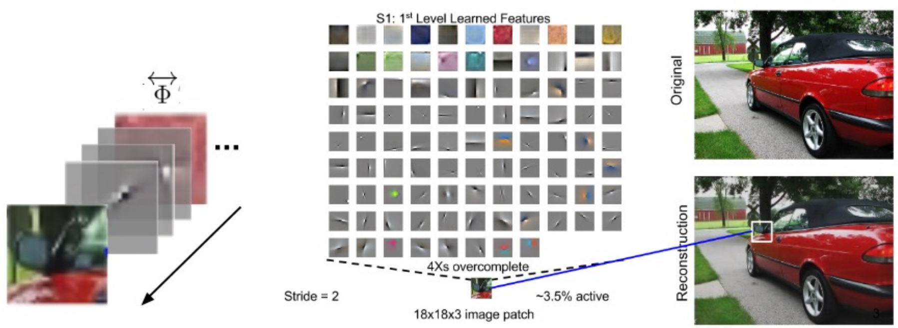
ANN Autoencoder Structure



CNN Autoencoder Structure
Encoder (top) and decoder (bottom)

PetaVision

- We also used PetaVision, a neuromorphic sparse coder
- Has been used for impressive computer vision problems
- Learned a Spatiotemporal Dictionary for predicting a frame given the two previous frames



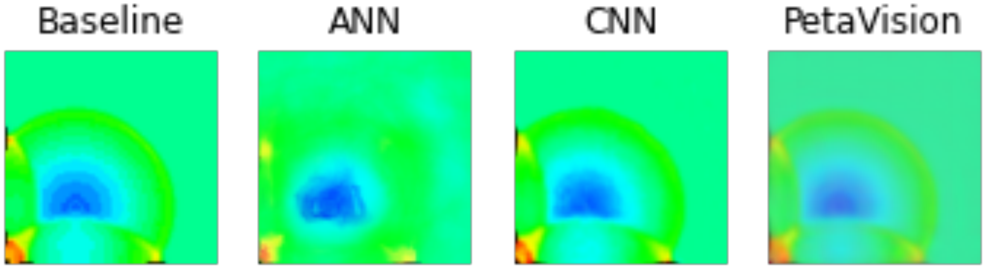
$$\overbrace{\overrightarrow{I}}^{\text{Original}} \approx \overbrace{\overleftrightarrow{\Phi}\, T(\overrightarrow{u})}^{\text{Reconstruction}}$$

$$E = \overbrace{\frac{1}{2}||\overrightarrow{I} - \{\overleftrightarrow{\Phi}\, T(\overrightarrow{u})\}||_2^2}^{\text{Reconstruction Term}} + \overbrace{\lambda||T(\overrightarrow{u})||_1}^{\text{Sparsity Term}}$$

Illustration of how PetaVision reconstructs images

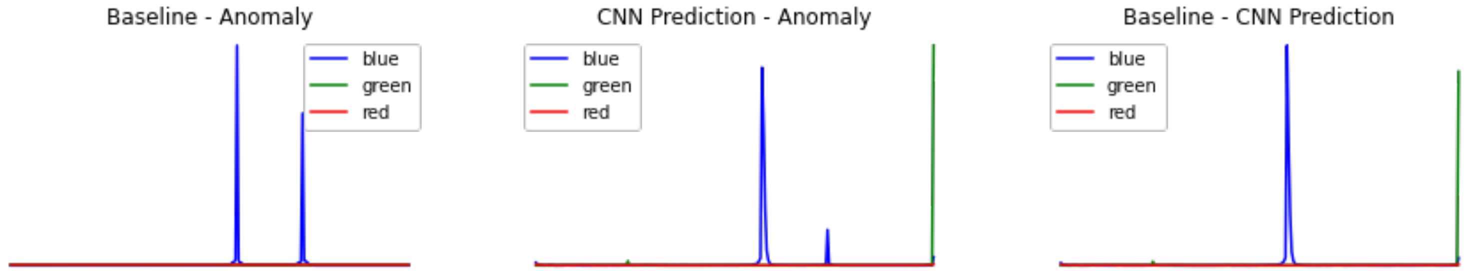
Predictions

- Each autoencoder achieved different levels of reconstruction
  - ANN failed to recreate the proper shapes, colors, and patterns
  - CNN achieved correct colors but the shapes were not sharp
  - PetaVision learned accurate shape reconstructions but the coloring was off



Classification

- In order to classify anomalous frames, we converted the images to feature vectors and computer the difference
- Analysis showed that there were often differences in the blue channel
- We found that one set of blue features in particular were found in the anomalies



Feature Vector Differences Represented as Histograms

Classification (cont)

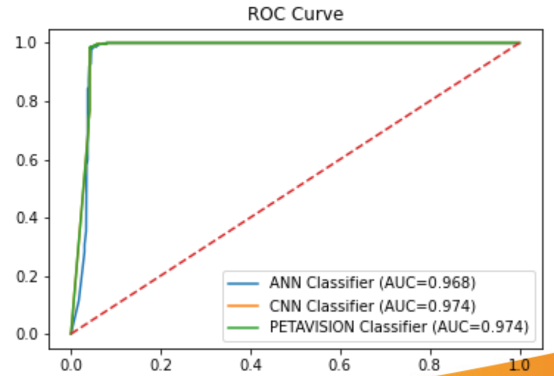
- We trained a Gradient Boosted Decision Tree (GBDT) Classifier
  - Also tried random forest and multi-layer perceptron but they performed worse
- We used the feature vector differences as input
  - 3 channels X 256 color values = 768 features
- Each Autoencoder was paired with its own classifier

Conclusions

Results

- Overall we got very similar classification performance regardless of autoencoder
- The classifier achieved impressive metrics (~0.97 AUC)
- This was surprising considering the difference in frame predictions

Auto-Encoder	Accuracy	Precision	Recall	F1	AUC
ANN	0.966	0.944	0.992	0.967	0.968
CNN	0.969	0.950	0.991	0.970	0.978
PetaVision	0.966	0.942	0.994	0.967	0.974



Conclusions

- Despite differences in predictions, we achieved very good classification results
- Using a random forest we discovered that only a small subset of features seemed to attribute to the classification
  - That blue spike seen in the histograms earlier
- Our dataset may have been too “easy” for the classification, but this is still a good first step

Next Steps

- Generate a new dataset of more complex simulations
- Look at more computer vision techniques
  - PetaVision has potential we did not use here
- Find different types of anomalies that may appear
- Trace an anomaly through timesteps if possible

References

[1] Sridharan, V., DeBardeleben, N., Blanchard, S., Ferreira, K. B., Stearley, J., Shalf, J., and Gurumurthi, S., “Memory errors in modern systems: The good, the bad, and the ugly,” in [Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems], ASPLOS ’15, 297–310, Association for Computing Machinery, New York, NY, USA (2015).
[2] Snir, M., Wisniewski, R. W., Abraham, J. A., Adve, S. V., Bagchi, S., Balaji, P., Belak, J., Bose, P., Cappello, F., Carlson, B., Chien, A. A., Coteus, P., DeBardeleben, N. A., Diniz, P. C., Engelmann, C., Erez, M., Fazzari, S., Geist, A., Gupta, R., Johnson, F., Krishnamoorthy, S., Leyffer, S., Liberty, D., Mitra, S., Munson, T., Schreiber, R., Stearley, J., and Hensbergen, E. V., “Addressing failures in exascale computing,” The International Journal of High Performance Computing Applications 28(2), 129–173 (2014).
[3] Guan, Q., DeBardeleben, N., Atkinson, B., Robey, R., and Jones, W. M., “Towards building resilient scientific applications: Resilience analysis on the impact of soft error and transient error tolerance with the clamr hydrodynamics mini-app,” in [2015 IEEE International Conference on Cluster Computing], 176–179 (2015).
[4] Yuan, Y., Wu, Y., Wang, Q., Yang, G., and Zheng, W., “Job failures in high performance computing systems: A large-scale empirical study,” Computers Mathematics with Applications 63(2), 365–377 (2012). Advances in context, cognitive, and secure computing.
[5] Chow, J., Su, Z., Wu, J., Tan, P., Mao, X., and Wang, Y., “Anomaly detection of defects on concrete structures with the convolutional autoencoder,” Advanced Engineering Informatics 45, 101105 (2020).
[6] Zhao, Q. and Karry, F., “Anomaly detection for images using auto-encoder based sparse representation,” Image Analysis and Recognition. ICIAR 2020. Lecture Notes in Computer Science 12132, 144–153 (2020).
[7] Lu, Y. and Xu, P., “Anomaly detection for skin disease images using variational autoencoder,” (2018).
[8] Zhao, Y., Deng, B., Shen, C., Liu, Y., Lu, H., and Hua, X.-S., “Spatio-temporal autoencoder for video anomaly detection,” in [Proceedings of the 25th ACM International Conference on Multimedia], MM ’17,
1933–1941, Association for Computing Machinery, New York, NY, USA (2017).
[9] Shin, W., Bu, S.-J., and Cho, S.-B., “3d-convolutional neural network with generative adversarial net- work and autoencoder for robust anomaly detection in video surveillance,” International Journal of Neural
Systems 30(06), 2050034 (2020). PMID: 32466693.
[10] “PetaVision.” https://petavision.github.io/. Accessed: 2021-07-12.
[11] Nicholaeff, D., Davis, N., Trujillo, D., and Robey, R. W., “Cell-based adaptive mesh refinement implemented
with general purpose graphics processing units,” (2012).
[12] Chollet, F. et al., “Keras,” (2015).
[13] Watkins, Y., Thresher, A., Mascarenas, D., and Kenyon, G. T., “Sparse coding enables the reconstruction
of high-fidelity images and video from retinal spike trains,” in [Proceedings of the International Conference
on Neuromorphic Systems], ICONS ’18, Association for Computing Machinery, New York, NY, USA (2018).
[14] Rozell, C. J., Johnson, D. H., Baraniuk, R. G., and Olshausen, B. A., “Sparse Coding via Thresholding and
Local Competition in Neural Circuits,” Neural Computation 20, 2526–2563 (10 2008).
[15] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., “Scikit-learn: Machine learning in Python,” Journal of Machine Learning Research 12,
2825–2830 (2011).

Thank You

Contact Info: Adam Good (agood@lanl.gov)