SAND2022-10133C

Sandia National Laboratories

Exceptional service in the national interest

Learning to Parameterize a Stochastic Process Using Neuromorphic Data Generation

William Severa, J. Darby Smith, Richard Lehoucq, and Brad Aimone

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline

- How to verify a scientific spiking neuromorphic algorithm?
- Our answer: Make it an inverse problem and use ML
- Our inverse problem: OU process simulated on Loihi
- Our ML method: Apply CNNs to image-like data
- Concluding thoughts

Expanding Neuromorphic Workloads

Spiking neuromorphic systems are being used in more and more domains.

- Real-time sensor processing
- Robotics control
- Scientific and numerical workloads

Expanding Neuromorphic Workloads

Spiking neuromorphic systems are being used in more and more domains.

- Real-time sensor processing
- Robotics control
- Scientific and numerical workloads

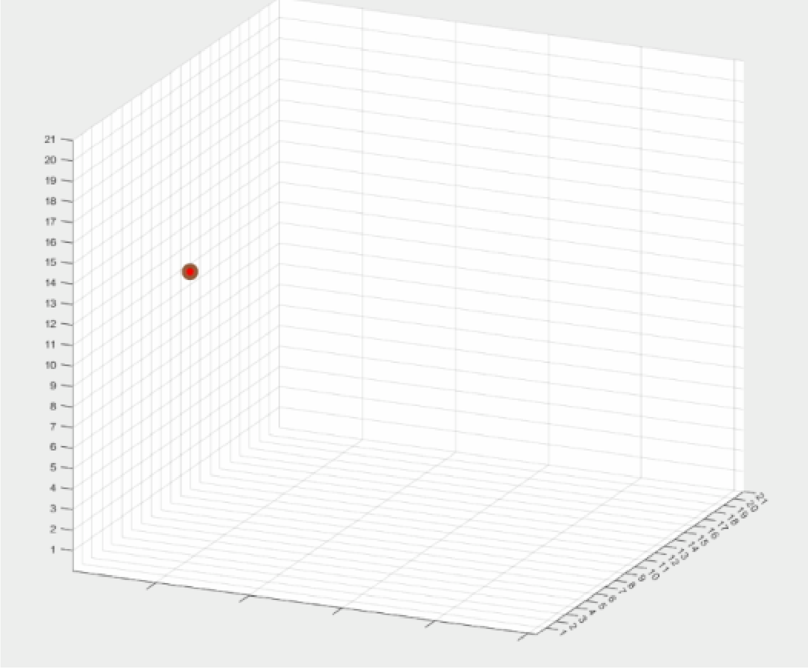
Evaluating performance

Ways to know if a scientific spiking neuromorphic algorithm is working:

- Classifier – Accuracy
- Regression – Absolute error
- Stochastic – ?
  - Quantity of Interest (QOI): You can compute what you care about
  - Statistical: You are close in distribution
  - Parameter Recovery: You can determine underlying system (Inverse Problem)

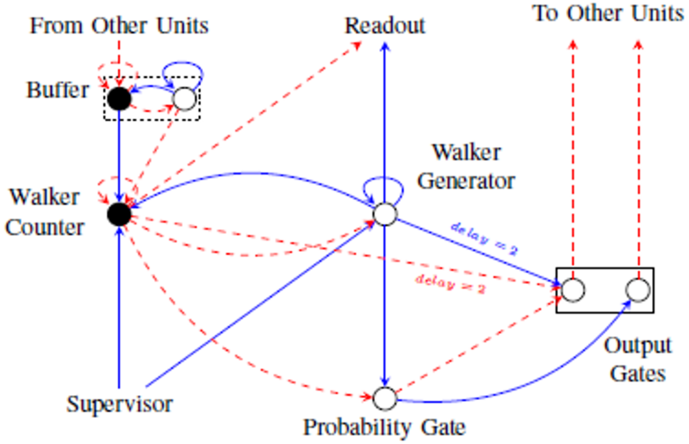
Random Walks On Neuromorphic Systems

- Previously, we developed spiking neuromorphic algorithms for random walks[1]
- Two main formulations:
  - A group of neurons represents a walker
    - Activity represents position
  - A group of neurons represents a location (Density Method)
    - Activity represents walkers



Simple diffusion on Loihi

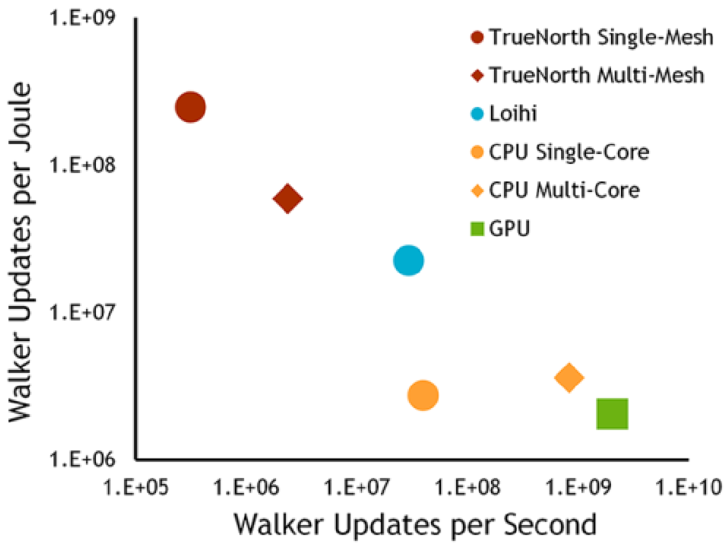
Example Density Circuit



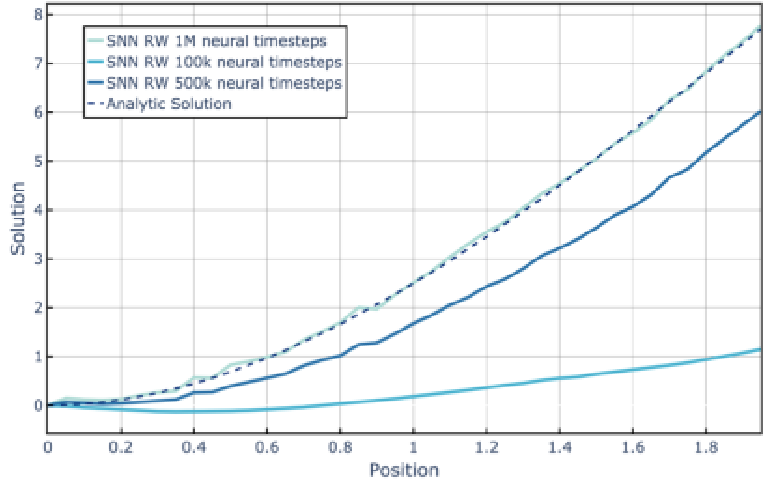
1 Severa, IJCNN, 2018

Density Based Method

- Nodes can be connected in arbitrary graphs and with arbitrary transition probabilities (depending on hardware)
- Walkers scale efficiently
- Requires discretization of underlying system
- Energy efficient solutions to a large family of SDEs[1]
- Applied to a steady state heat equation on IBM TrueNorth and Intel Loihi[2]
- Several examples of good QOI estimation



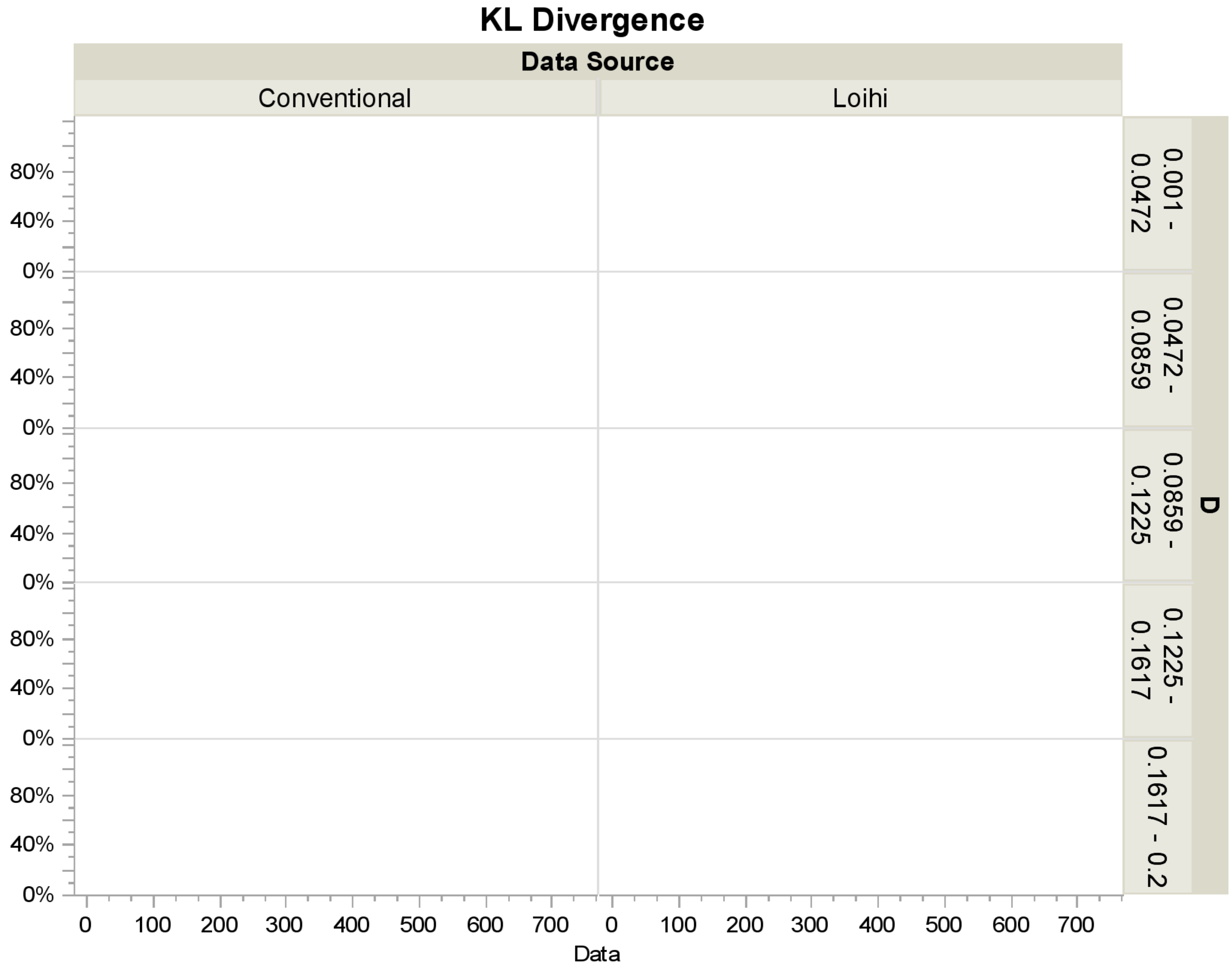
QOI Error Decreasing with Runtime



1 Smith, Nat. Elec., 2022
2 Smith, ICONS, 2020

Density Based Method Generates Statistically Similar Samples

- One way to validate the method is characterize the distribution of generated samples[1]
- Various statistical distances exist
  - Log Likelihood Ratio
  - KL Divergence
- Some of the Loihi samples deviate from expected
- Vast majority are pretty close
- Is there another way to verify that the samples are useful?



1 Aimone, ICRC, 2021

Inverse problem means recovering underlying parameters

- Inverse problems are finding underlying parameters from observations
- Applications in many experimental domains
- Solving an inverse problem from simulated data means the data is ‘useful’

Our inverse problem

We focus on a 1D Ornstein-Uhlenbeck (OU) process:

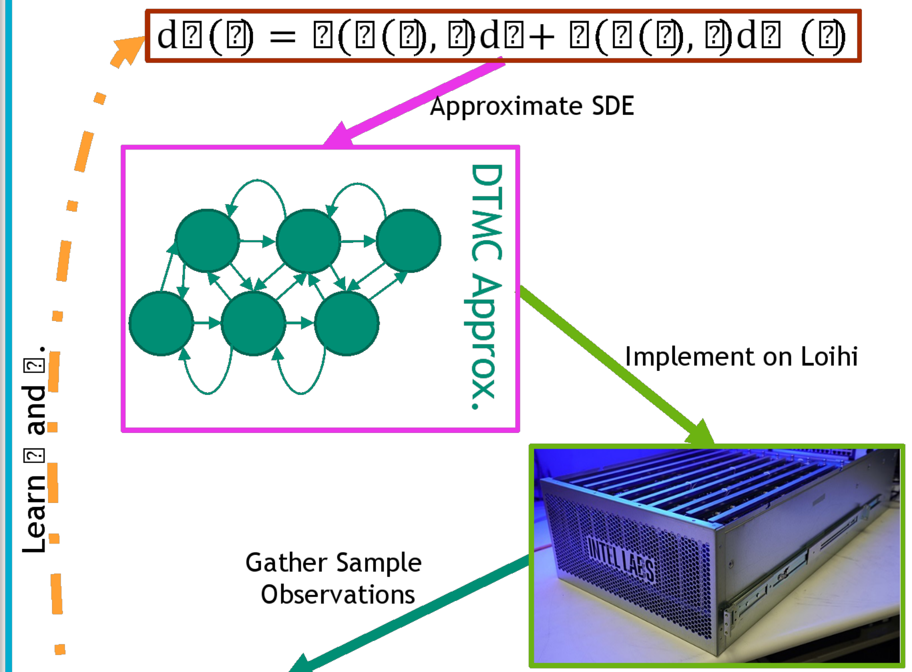
$$X(t) = X(0) - k\int_0^t (X(u) - z)\,\mathrm{d}u + \sqrt{2D}W(t)$$

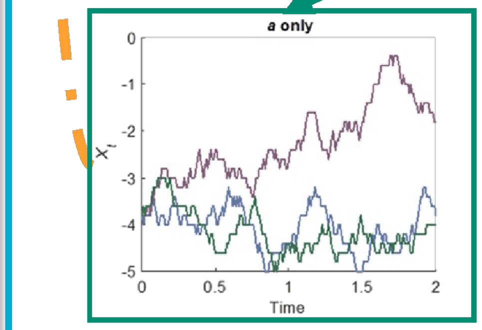
Our goal will be to recover the parameters $\boldsymbol{k}$, $\boldsymbol{D}$, and $\boldsymbol{z}$.

This OU process has applications in molecular motor motion, stock prices, thermal diffusive particle in a harmonic well, epidemiological processes, and more.

Setup Overview

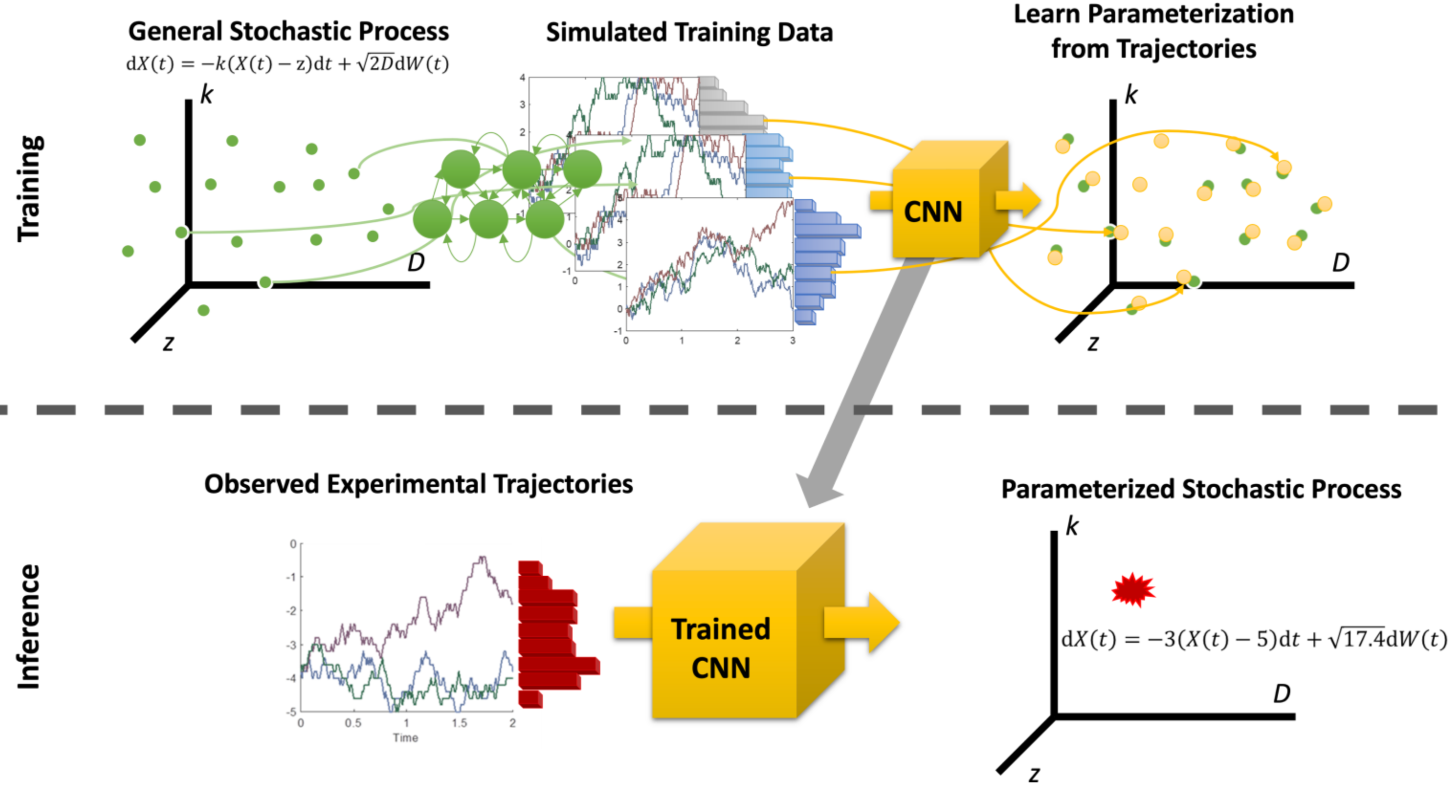
Data Generation (Loihi)



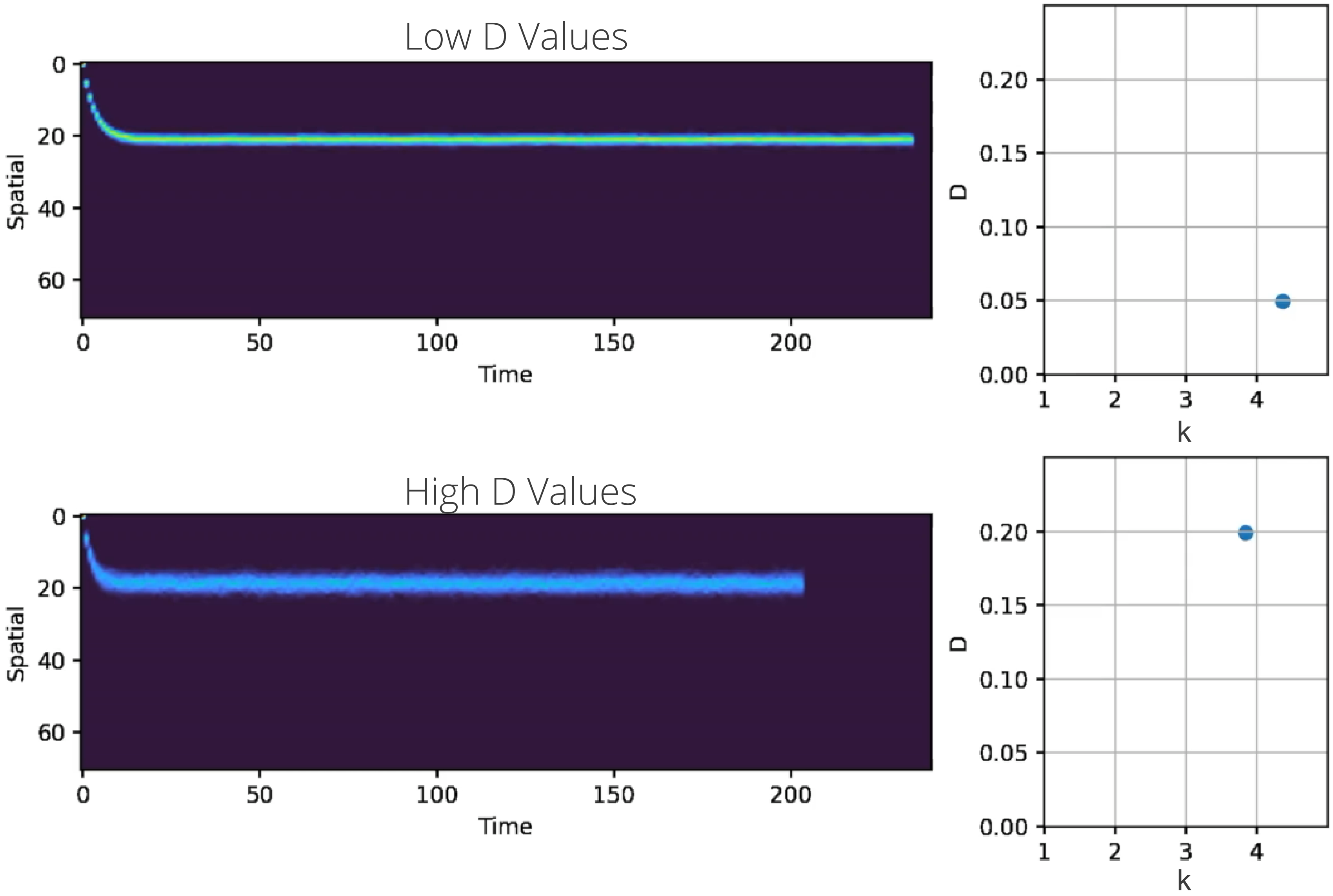


We may sample paths of an SDE by simulating them through an approximating DTMC. The DTMC is well suited for implementation on Loihi, where sample observations can be generated. **We aim to use machine learning to** help solve the inverse problem, finding □ and □ when only sample observations are available.

Network Training (GPU)



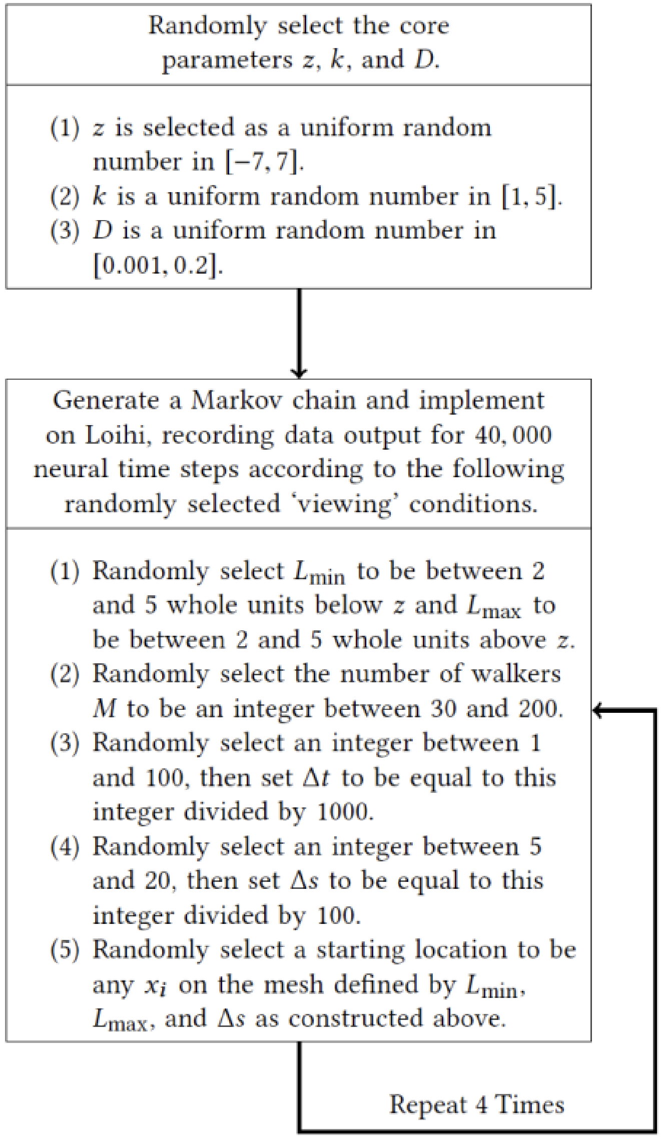
Examples from generated data (Brighter = More Walkers)



Dataset Overview

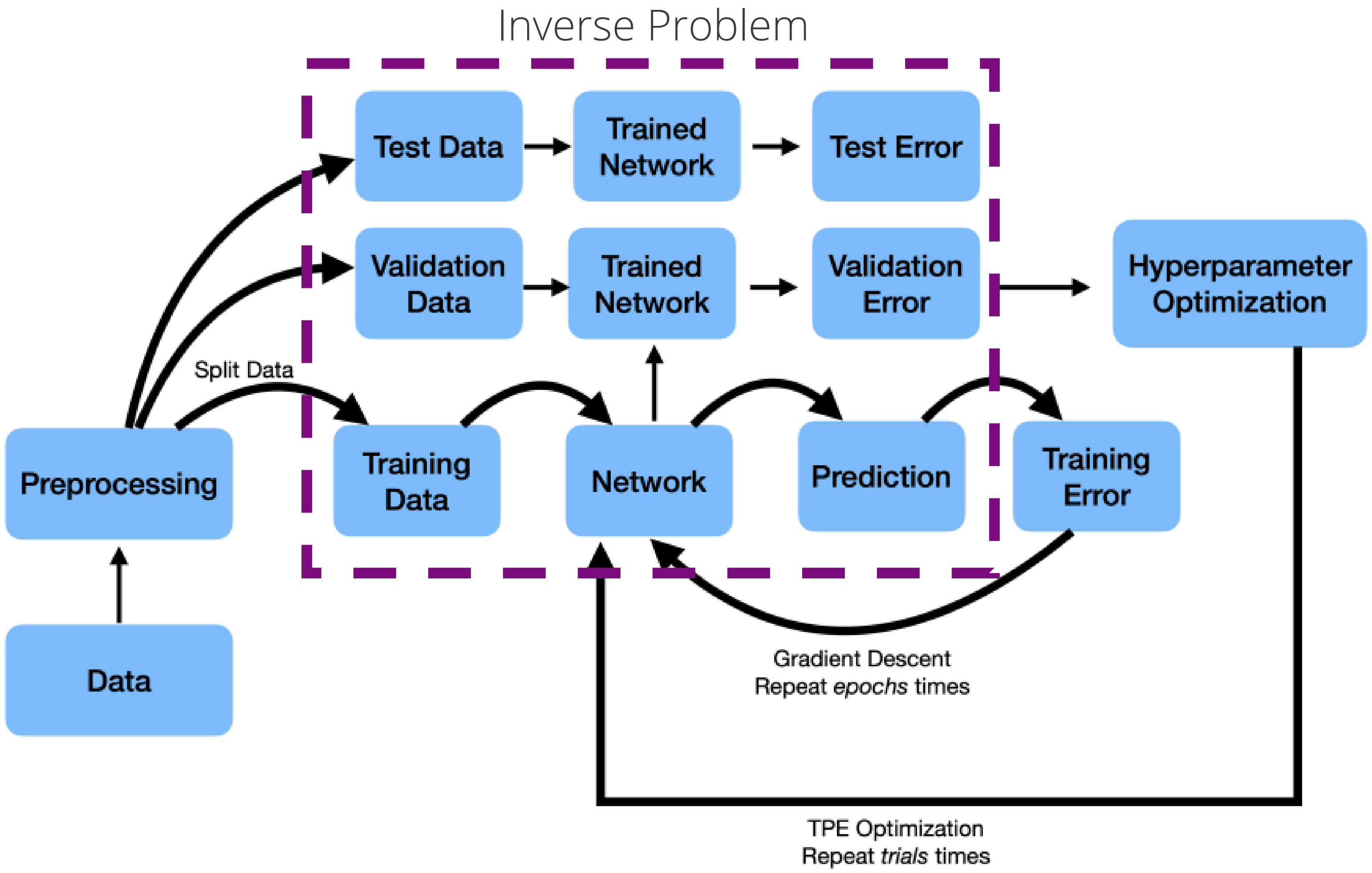
- Generate data and section into two datasets:
  - “Base” 25,874 samples
  - “Expanded” 37,554 samples
- Validation set 3697 samples
- MATLAB generated “Conventional” data, 4163 samples
- Spatial x Time means data is image-like

Figure 2: Process for generating DTMC on Loihi.

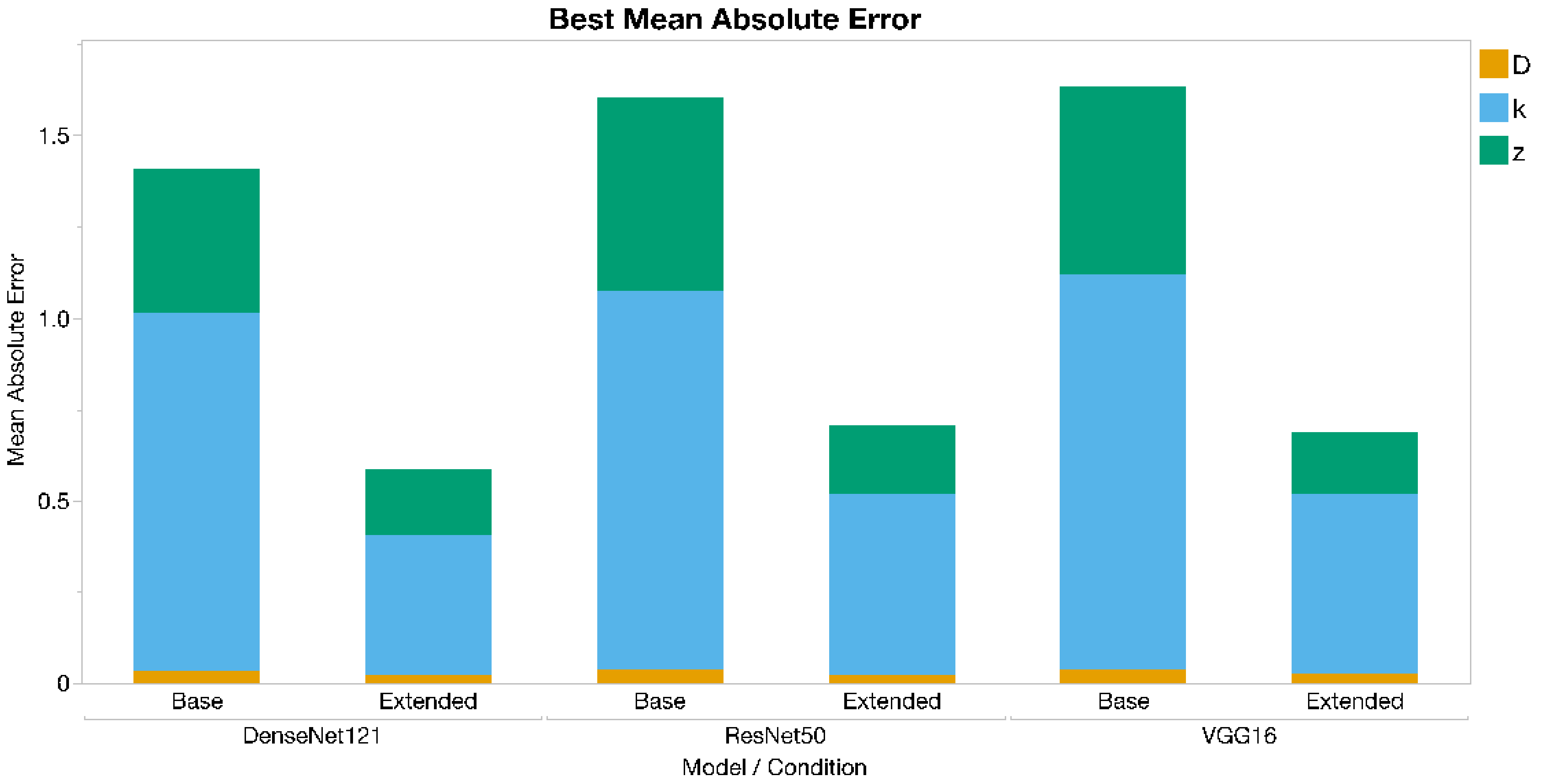


Setup Overview

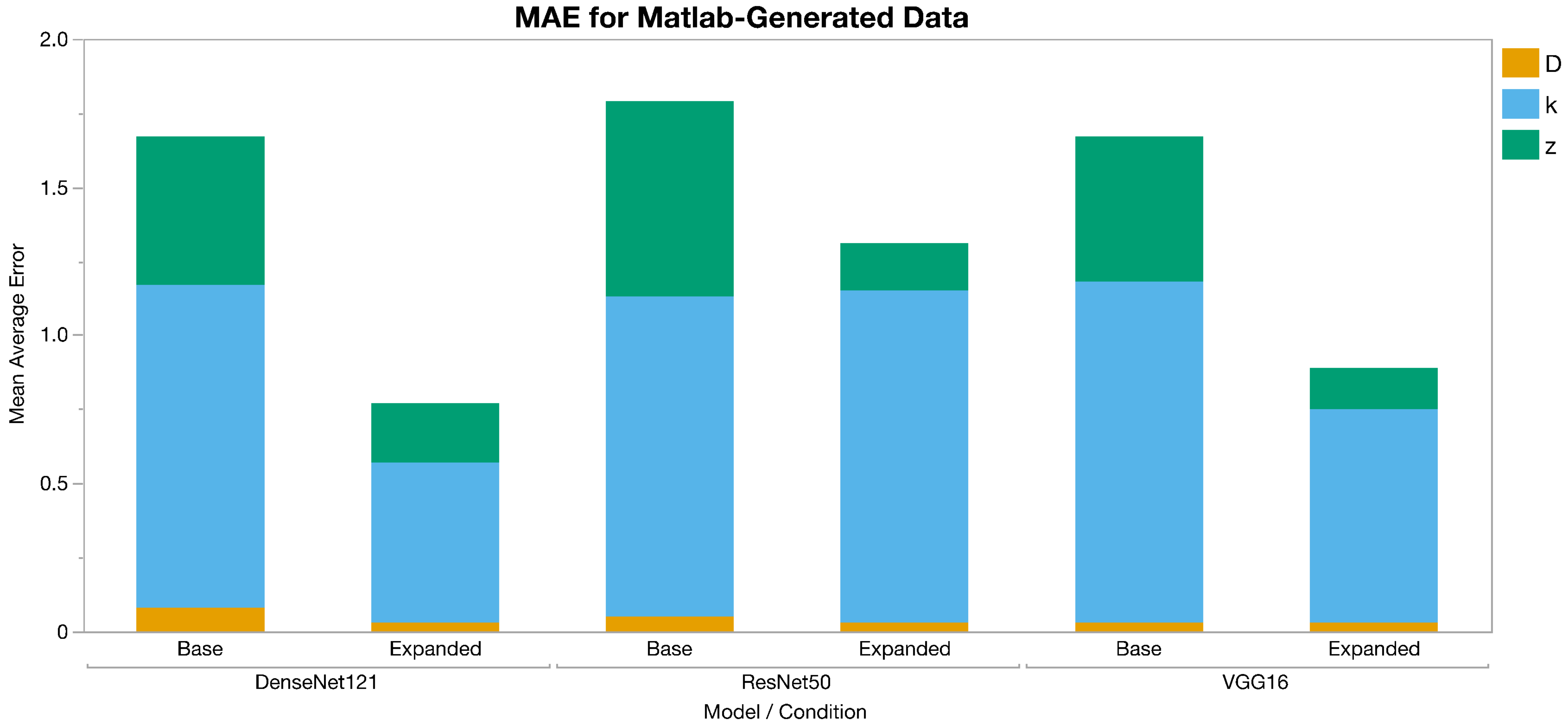
- Studied 3 off-the-shelf CNNs:
  - ResNet50
  - DenseNet121
  - VGG16
- TPE hyperparameter optimization using Optuna
- Zero padding
- Squared Error for loss
- Mean Absolute Error (MAE) as reported metric
- *No domain knowledge used*



Results (Validation)



Evaluation on MATLAB Test Set



Loss During Training

Loss During Training

Training losses (especially for VGG) are pretty good.

Loss During Training

Validation dropoff suggests more data needed.

Conclusion and Next Steps

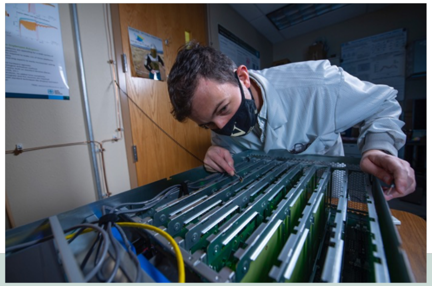
- Solving an inverse problem is neat
  - Generating simulation data to feed GPU-based training (Heterogeneous workload)
- Method to validate a spiking neuromorphic algorithm
  - Losses are low (though could be improved)
  - Off-the-self CNNs worked fine (though could be overparameterized)
- Implication: Energy efficient generation of simulation data
  - Scientific deep learning (and really all of deep learning) needs many samples
  - In many applications, that data is generated in simulators
  - Moving the simulation to neuromorphic could mean an energy savings
- Suggestion: First step for a fully neuromorphic approach
  - Simulations and learning on-neuromorphic
  - Use learning as a constraint/regularizer

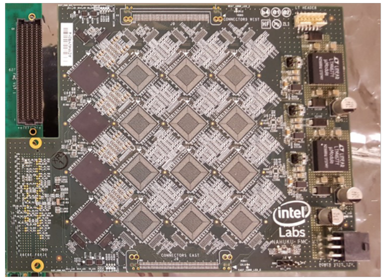
Thanks

- Co-Authors: Darby Smith, Brad Aimone, Rich Lehoucq
- Walker Rickord for providing editorial comments on the manuscript
- Craig Vineyard and Intel

Loihi Deployments at Neural Exploration and Research Lab (NERL)

1M Neurons
- FY19
- 8 Loihi Chips





50M Neurons
- FY20
- 384 Loihi Chips

128M Neurons
- FY22
- 2nd Gen Arch

1B Neurons
- FY23
- 2nd Gen Arch