SAND2024-07383C

Exceptional service in the national interest

GRAPH CONVOLUTIONAL NEURAL NETWORKS AS SURROGATE MODELS FOR CLIMATE SIMULATION

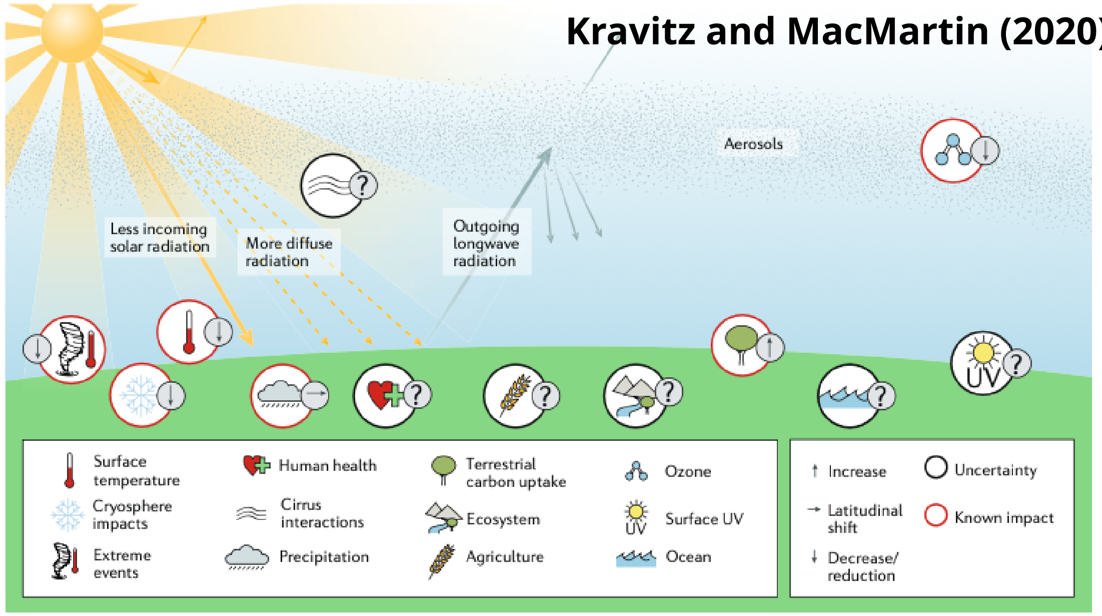
Performance Analysis for Climate Intervention (PACI)

Kevin Potter, Carianne Martinez, Samantha Brozak, Steven Sleder, Lauren Wheeler

STRATOSPHERIC AEROSOL INJECTION (SAI) WILL HAVE WIDESPREAD IMPACTS

- Scenarios for SAI are typically simulated in ESM (Earth System Models) and assessed for impacts on the climate system
- Impacts vary due to injection location (altitude and latitude), injection magnitude, injection timing, and the baseline emissions scenario
- *Decision makers or regulators will need scenario-specific information on impacts*
- This will require many more simulations (and assessments) of different SAI scenarios
- Surrogate models may be a potential tool for rapid assessment

Kravitz and MacMartin (2020)



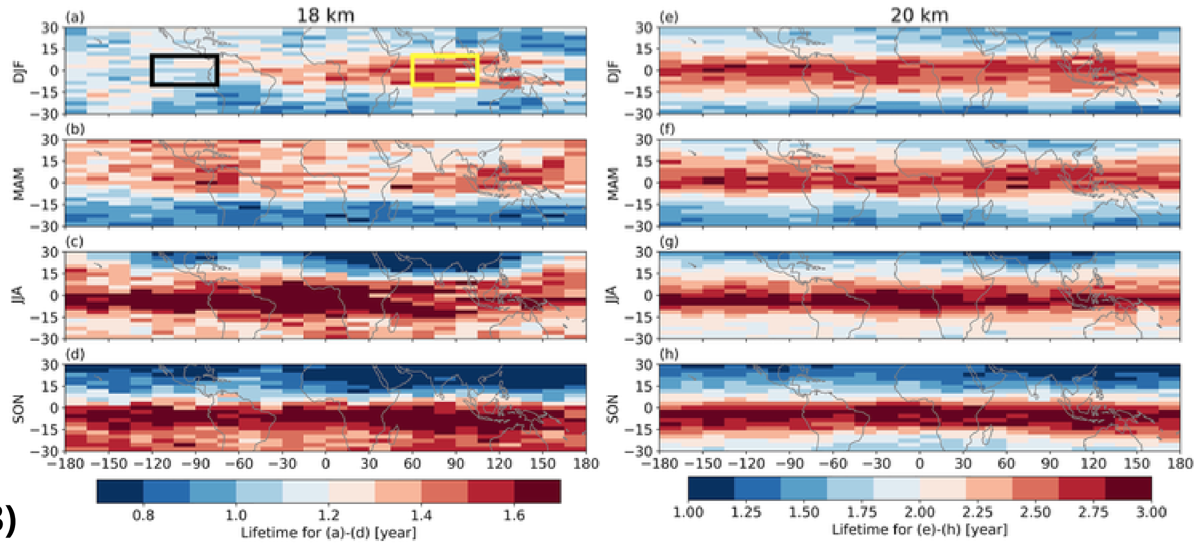


Figure 1. Particle lifetime distribution as a function of initial injection longitudes and latitudes.

Sun et al. (2023)

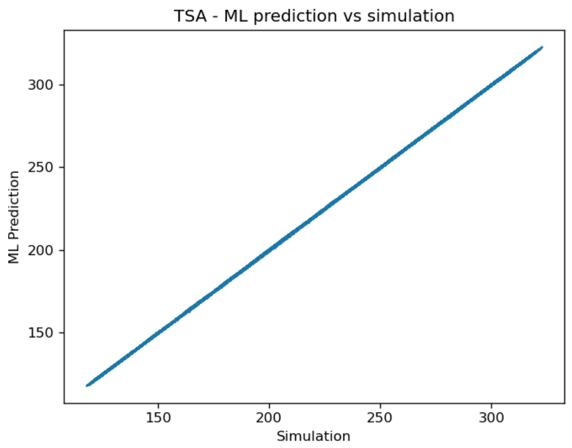
WHY DO WE NEED SURROGATE MODELS?

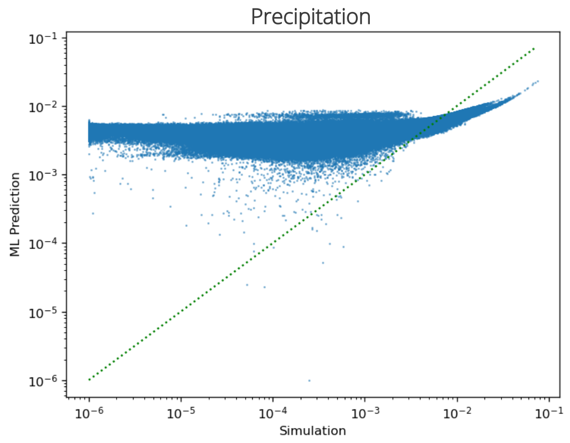
- Seeking to apply Performance Analysis (PA) to SAI:
  - Provides quantifiable measures of long term effects
  - Risk-risk model
- Need for Advanced Earth System Models (ESMs):
  - Critical to understand and predict global climate change impacts.
  - Traditional ESM simulations are time-intensive, taking weeks to run on large clusters.
  - PA requires thousands (or more) runs for proper uncertainty estimation
- Development of Rapid Prediction Simulations:
  - Focus on surrogate modeling to vet scenarios before extensive ESM simulation.
  - Aim to acquire climate impacts of SAI scenarios rapidly for preliminary analysis.
- Utilizing Graph Convolutional Neural Networks (GCNNs):
  - GCNN surrogate model predicting monthly temperature changes and other climate variables.
  - Enables multiple simulation runs to estimate uncertainties and possible impacts of SAI.
  - Performance and Efficiency of the Surrogate Model:
    - Capable of simulating 141 months in under 2 minutes using a single V100 GPU.
    - Simulate approximately 1000 months (90 years) in under an hour.

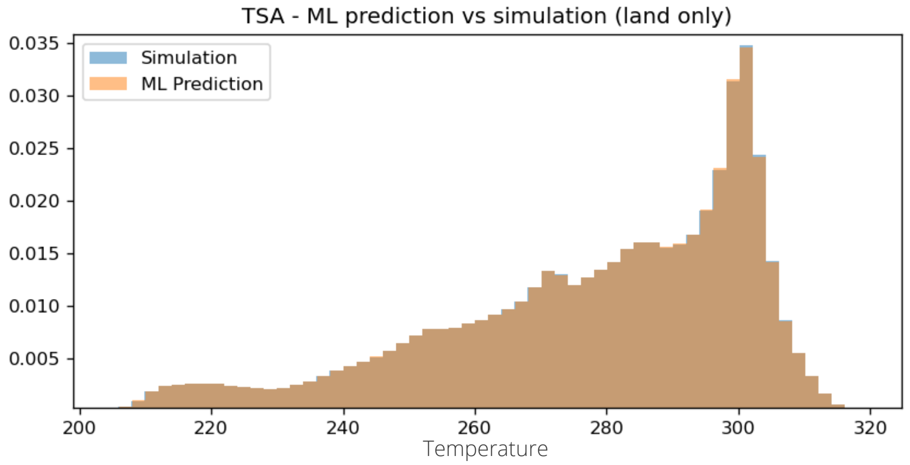
GCNNs allow thousands of simulations on a single GPU in the timeframe of a single traditional ESM run

TL;DR

- **Accuracy and Testing of the Model:**
  - **Tested using a holdout set from 4 separate GLENS control simulations of 141 months.**
  - **Mean Absolute Error (MAE) for temperature predictions below 0.5 degrees Kelvin**
  - **Maximum absolute error below 2 degrees Kelvin**
  - **Precipitation has challenges from the extreme range (6 orders of magnitude)**
  - **Prediction distributions matches simulation outputs very well**







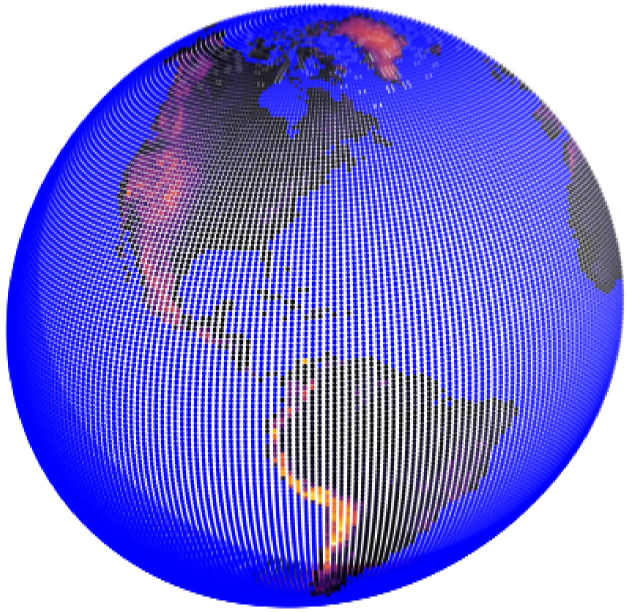
Graph CNNs can do very well with proper tuning

GLENS AND PREPROCESSING

- **Used data from Geoengineering Large Ensemble Project (GLENS)** [1]
  - **Extracted variables of interest to CSVs**
  - **Deal with missing data (e.g. TSA has no values over ocean)**
  - **Add a time in months**
  - **Changed PRECT to average per month**
  - Added past month's outputs (allow autoregressive prediction)

- **Graph specific**
  - **Convert from lat/lon to x, y, z**
  - **Downsample to reduce oversampling at poles**
  - **Add 4 edges between nearest neighbors**

- **Normalization done inline by model**



[1] https://www.cesm.ucar.edu/community-projects/glens

FULLY CONNECTED NEURAL NETWORK (FCNN) MODEL

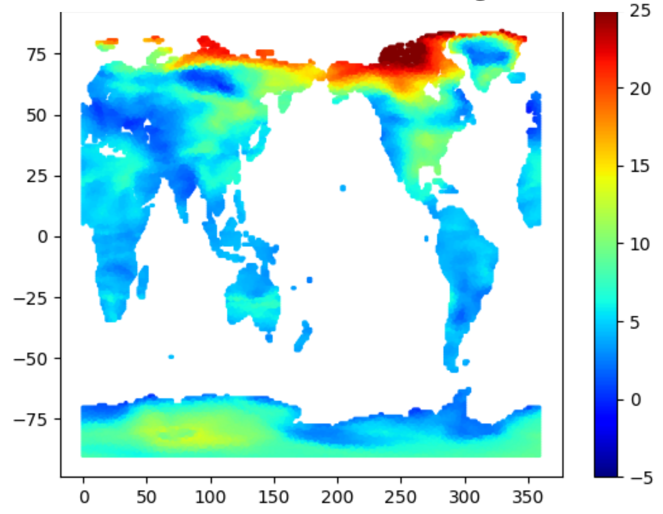
Input: year, month, latitude, longitude, aerosol

Output: Difference in TSA between control and feedback run

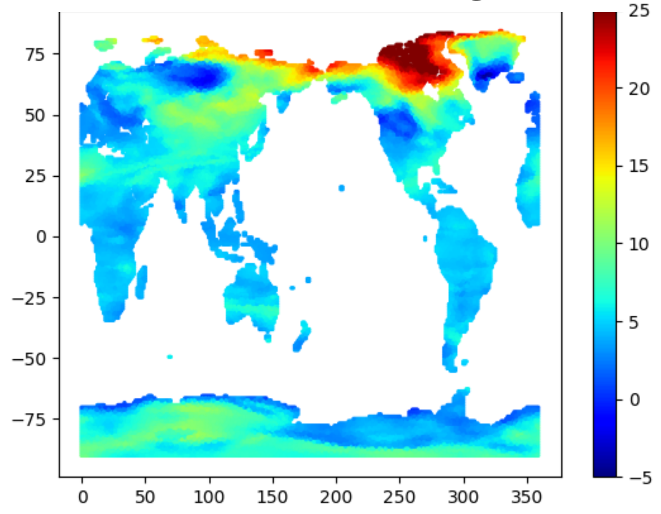
Training set: ~20-60 million points from GLENS simulations

Test set: Data from held out GLENS simulation

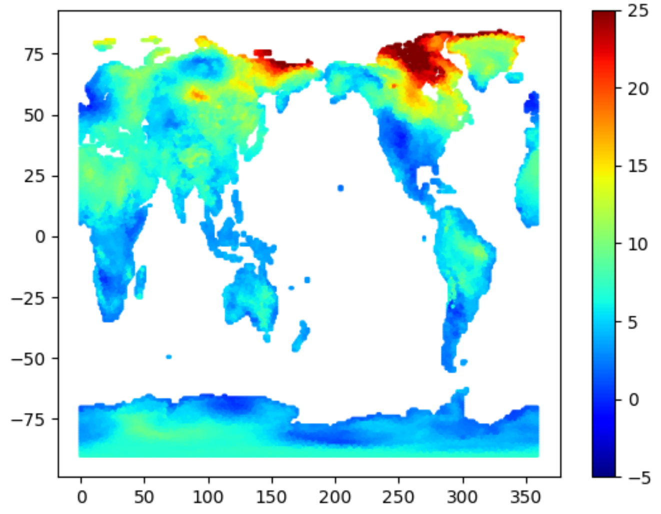
Predicted TSA diff
1 ensemble member training data



Predicted TSA diff
3 ensemble members training data

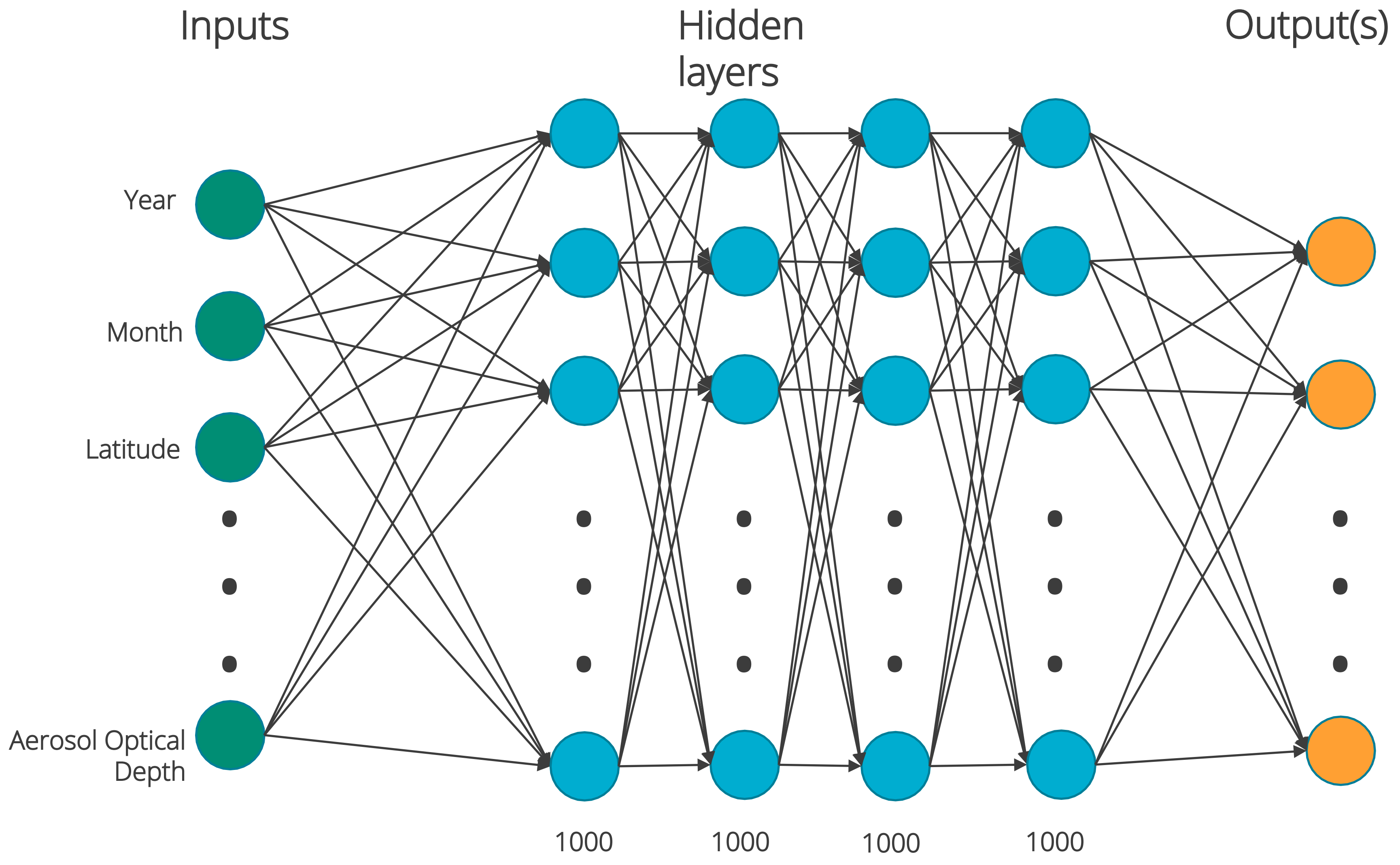


GLENS TSA
(Control run – feedback run)

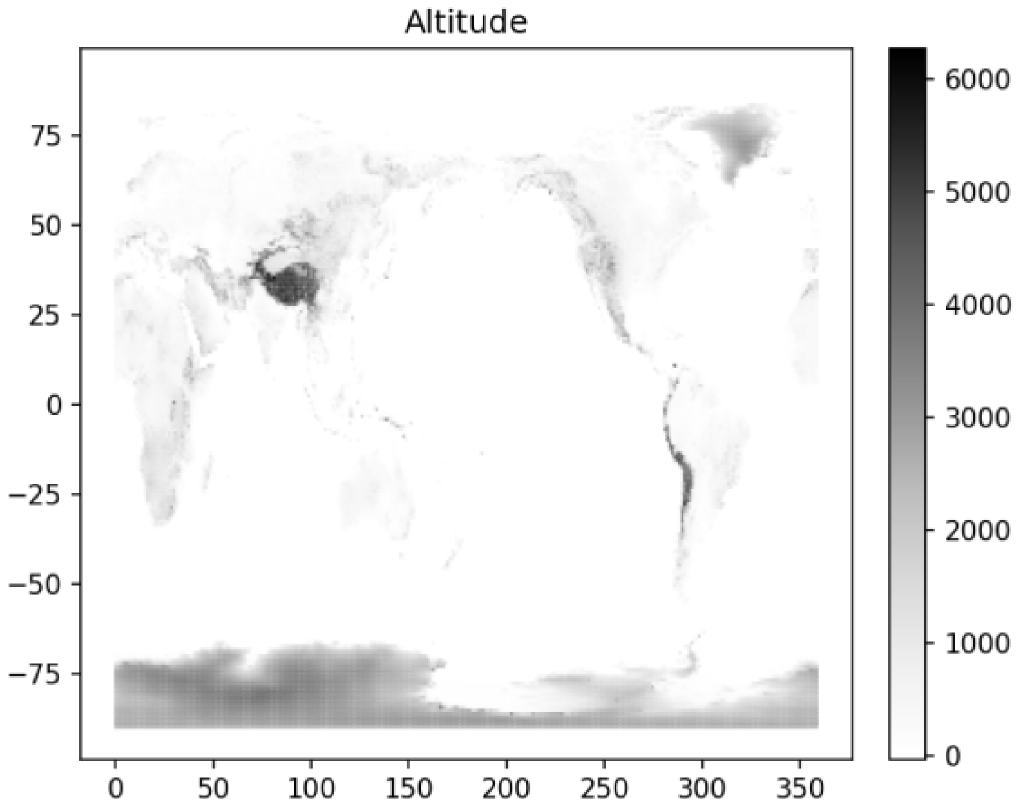


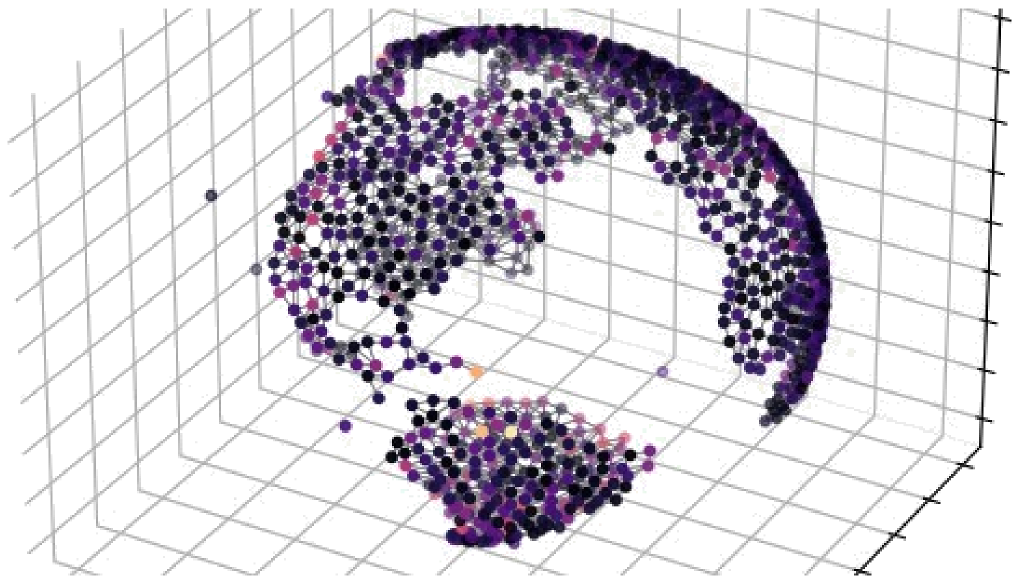
A fully connected neural network (NN) learned to predict the difference in surface temperature (TSA) over time between GLENS control runs and feedback runs with aerosol injection intervention. The plots above show a snapshot in time (January 2080) from the models. (Left) NN prediction given only one GLENS run as training data. (Center) NN prediction given 3 GLENS runs as training data. (Right) GLENS simulation output for the same month.

FCNN NETWORK ARCHITECTURE



WHY USE GRAPH CNNS?





3-D graph representation of surface temperature error downsampled (TSA).

(left) Scatterplot of the original data. Map projection causes an over-emphasis on the polar regions.
(right) No over sampling at polar regions and edges are connected in the now spherical geometry.

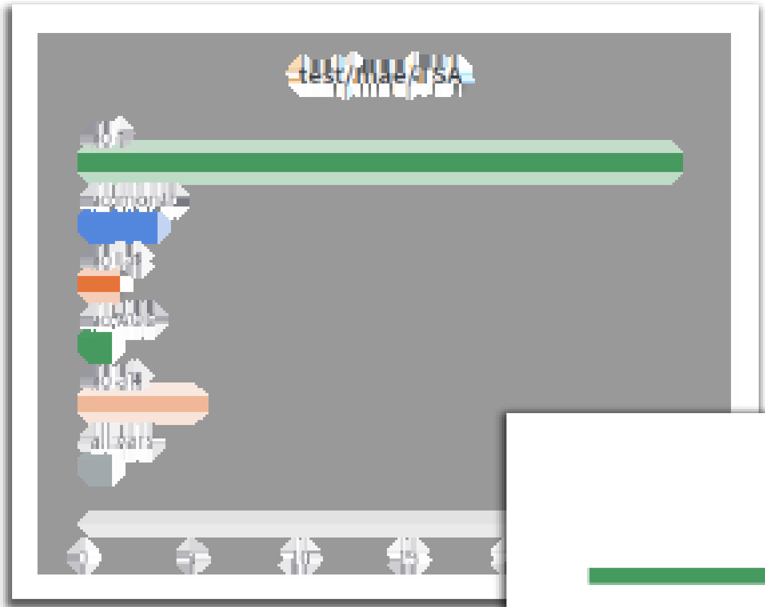
FCNN VS GCNN

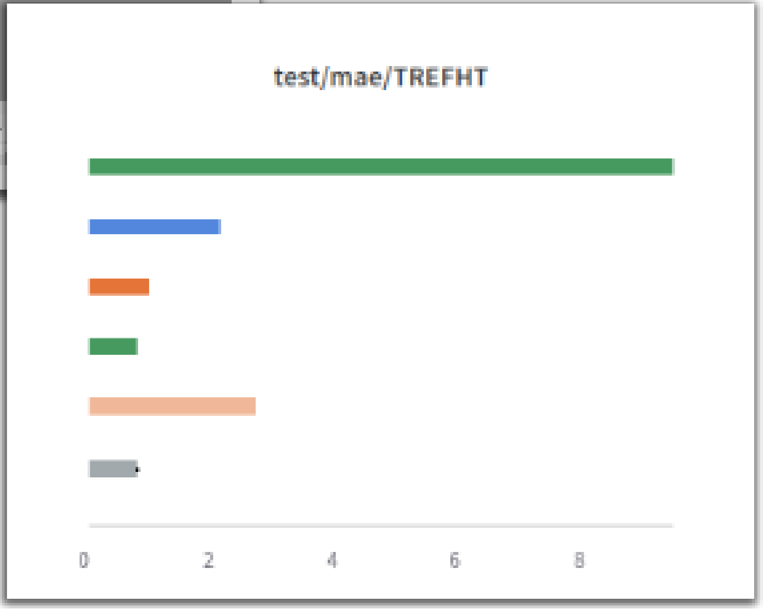
	Model	ALTMAX	ICEFRAC	PRECT	SNOWHLND	TREFHT	TSA
MAE	FCN	0.4611	0.1504	0.0012	0.1238	5.7008	21.7042
	GCNN	0.0341	0.0019	0.0010	0.0038	0.0707	0.0837
	FCN/GCNN	13.5266	81.1539	1.2776	32.5978	80.6600	259.2910
MARE	FCN	0.0163	131.2807	0.6694	107.9613	0.0409	0.1306
	GCNN	0.0010	0.9655	0.4658	2.7821	0.0005	0.0005
	FCN/GCNN	15.7740	135.9746	1.4370	38.8063	81.6119	289.3921
MaxAE	FCN	15.3309	1.3128	0.0736	0.8411	29.4854	84.0491
	GCNN	0.6211	0.1658	0.0697	0.1883	1.7796	1.9909
	FCN/GCNN	24.6842	7.9177	1.0545	4.4657	16.5684	42.2173
MaxARE	FCN	0.8522	485.1506	3.3429	427.6383	0.2321	0.6003
	GCNN	0.0347	86.1159	3.2700	126.1915	0.0171	0.0126
	FCN/GCNN	24.5861	5.6337	1.0223	3.3888	13.5839	47.4587

INPUT VARIABLE IMPORTANCE

Ran through with each input variable held out

- *t* **was unsurprisingly the most influential (no other variable has a relation to CO2 level for this test)**
- *Altitude* **was next most influential**
- *Month* **was a surprise though since month is recoverable from**
- *AOD* **was not important for this test (expected since this is just against the control dataset)**

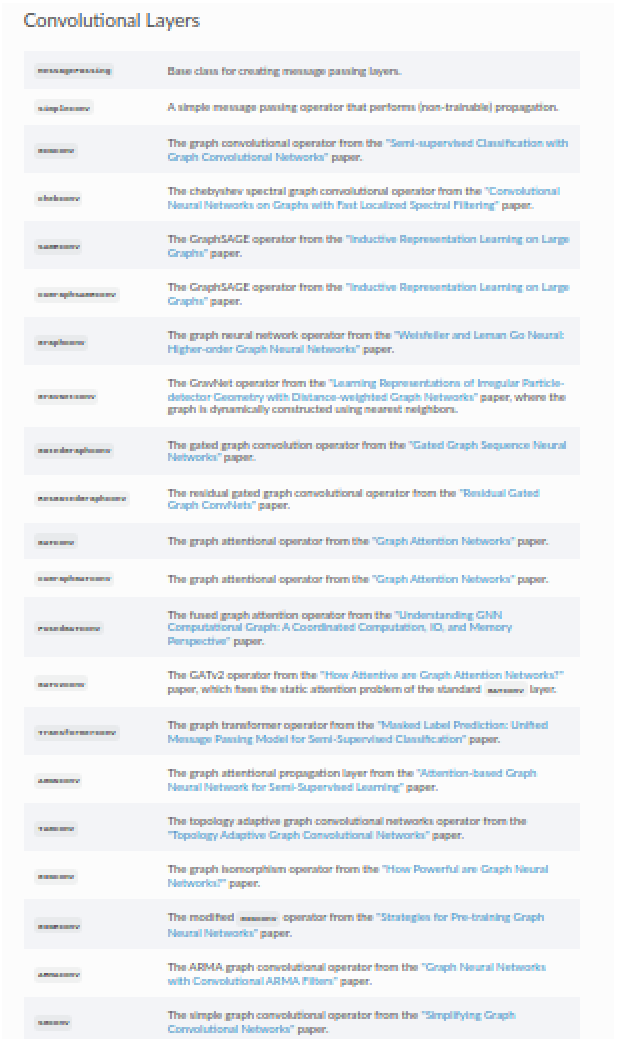




GCNN LAYER EVALUATION

There are dozens of graph convolutional layers available in PytorchGeometric

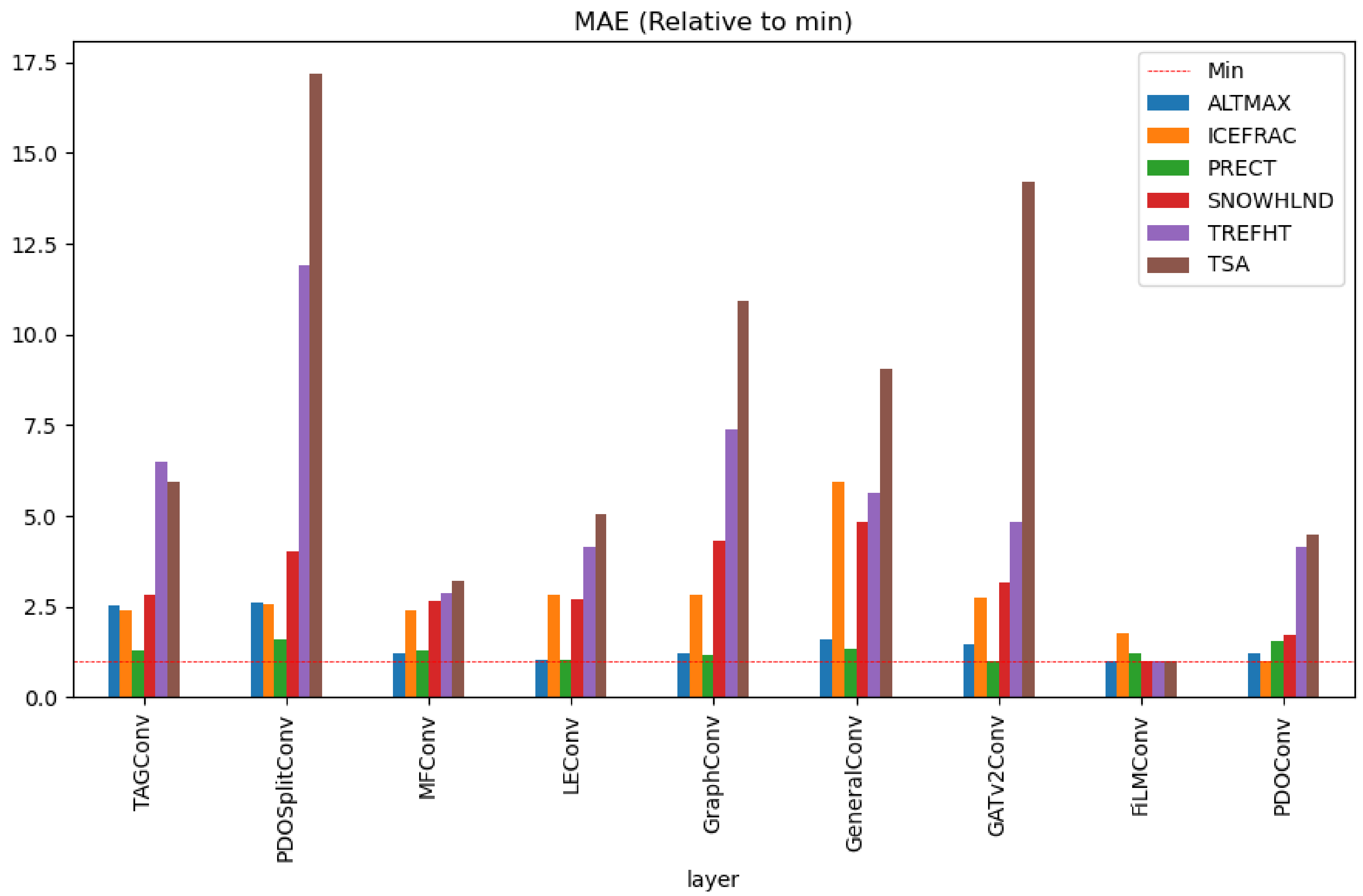
- **Tested the 20 that were easily applied**
- **Trained for 100 epochs**
- **Results for the best 9**
- **Metrics:**
  - **Mean Absolute Error (MAE)**
  - **Max Absolute Error (Max AE)**
  - **Measured relative to minimum error (of the layers tested)**


Convolutional Layers

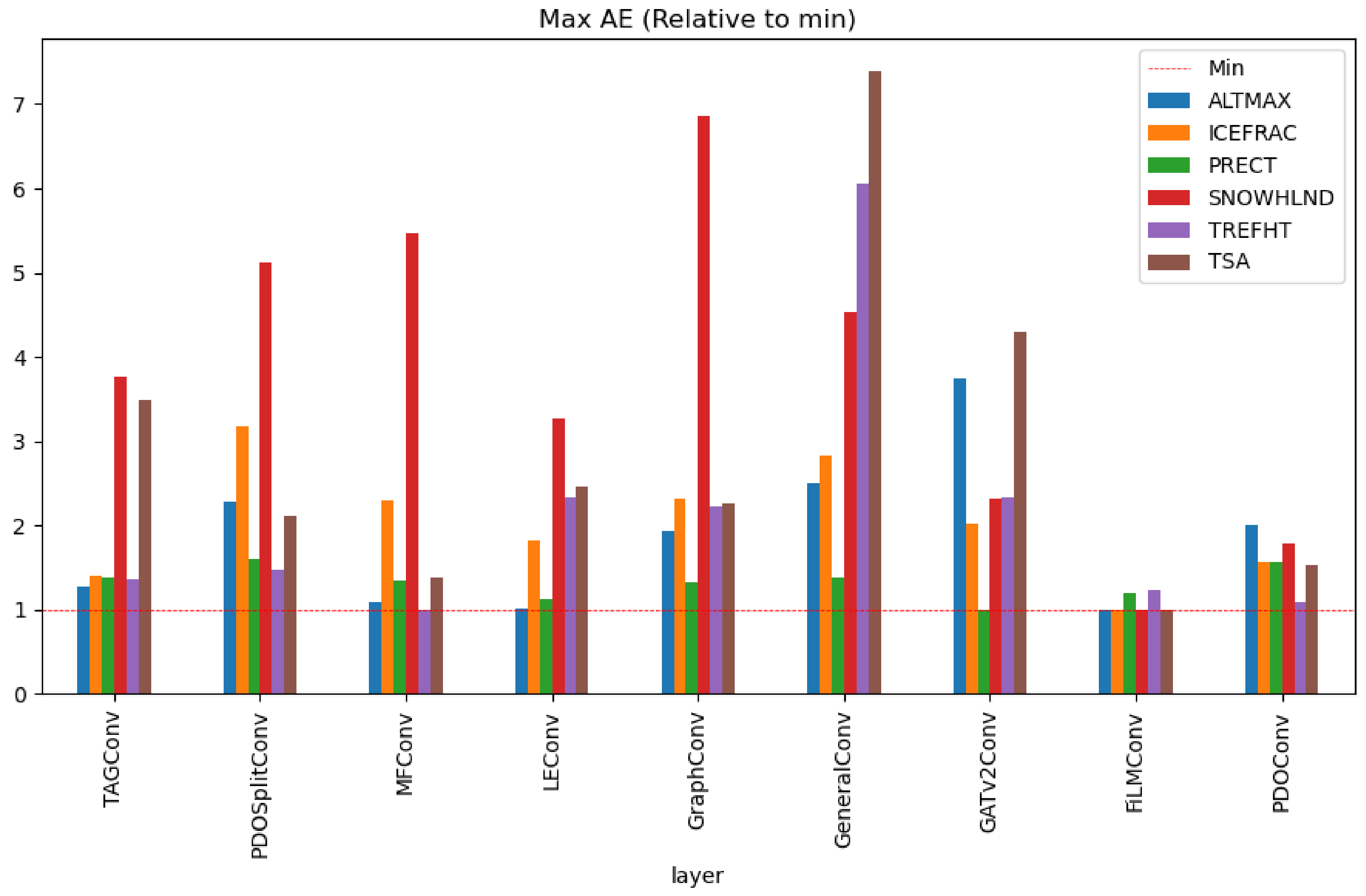
`MessagePassing`	Base class for creating message passing layers.
`SimpleConv`	A simple message passing operator that performs (non-trainable) propagation.
`GCNConv`	The graph convolutional operator from the "Semi-supervised Classification with Graph Convolutional Networks" paper.
`ChebConv`	The chebyshev spectral graph convolutional operator from the "Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering" paper.
`SAGEConv`	The GraphSAGE operator from the "Inductive Representation Learning on Large Graphs" paper.
`CuGraphSAGEConv`	The GraphSAGE operator from the "Inductive Representation Learning on Large Graphs" paper.
`GraphConv`	The graph neural network operator from the "Weisfeiler and Leman Go Neural: Higher-order Graph Neural Networks" paper.
`GravNetConv`	The GravNet operator from the "Learning Representations of Irregular Particle-detector Geometry with Distance-weighted Graph Networks" paper, where the graph is dynamically constructed using nearest neighbors.
`GatedGraphConv`	The gated graph convolution operator from the "Gated Graph Sequence Neural Networks" paper.
`ResGatedGraphConv`	The residual gated graph convolutional operator from the "Residual Gated Graph ConvNets" paper.
`GATConv`	The graph attentional operator from the "Graph Attention Networks" paper.
`CuGraphGATConv`	The graph attentional operator from the "Graph Attention Networks" paper.
`FusedGATConv`	The fused graph attention operator from the "Understanding GNN Computational Graph: A Coordinated Computation, IO, and Memory Perspective" paper.
`GATv2Conv`	The GATv2 operator from the "How Attentive are Graph Attention Networks?" paper, which fixes the static attention problem of the standard `GATConv` layer.
`TransformerConv`	The graph transformer operator from the "Masked Label Prediction: Unified Message Passing Model for Semi-Supervised Classification" paper.
`AGNNConv`	The graph attentional propagation layer from the "Attention-based Graph Neural Network for Semi-Supervised Learning" paper.
`TAGConv`	The topology adaptive graph convolutional networks operator from the "Topology Adaptive Graph Convolutional Networks" paper.
`GINConv`	The graph isomorphism operator from the "How Powerful are Graph Neural Networks?" paper.
`GINEConv`	The modified `GINConv` operator from the "Strategies for Pre-training Graph Neural Networks" paper.
`ARMAConv`	The ARMA graph convolutional operator from the "Graph Neural Networks with Convolutional ARMA Filters" paper.
`SGConv`	The simple graph convolutional operator from the "Simplifying Graph Convolutional Networks" paper.

https://pytorch-geometric.readthedocs.io/en/latest/modules/nn.html#convolutional-layers

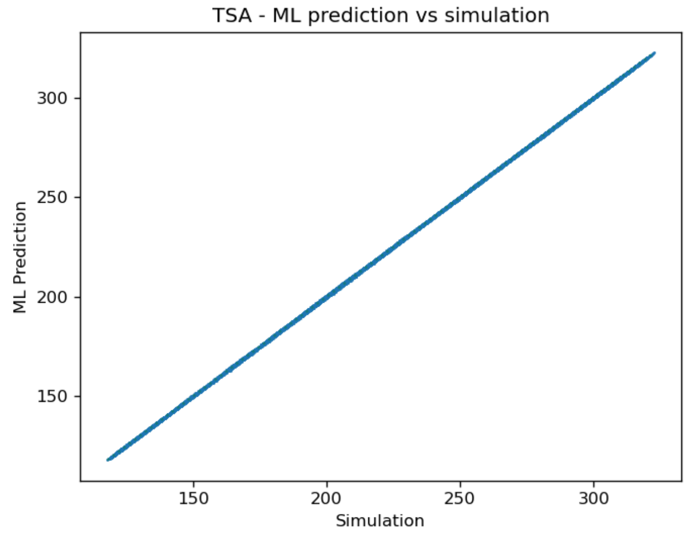
LAYER TESTS – RELATIVE MEAN ABSOLUTE ERROR

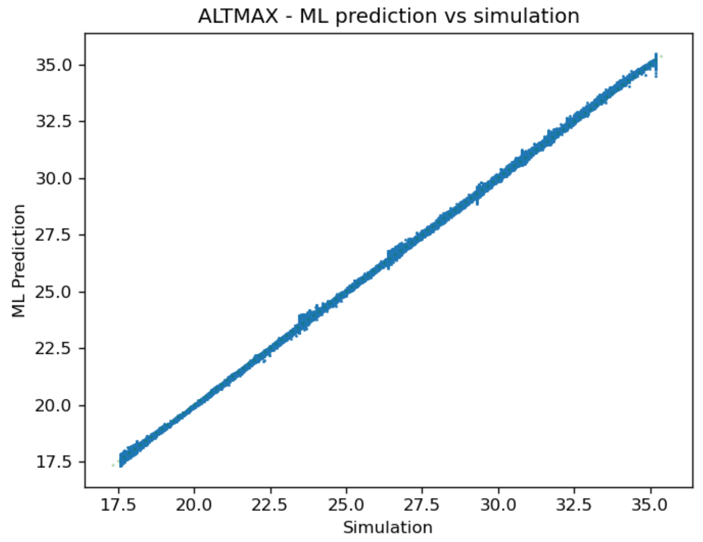


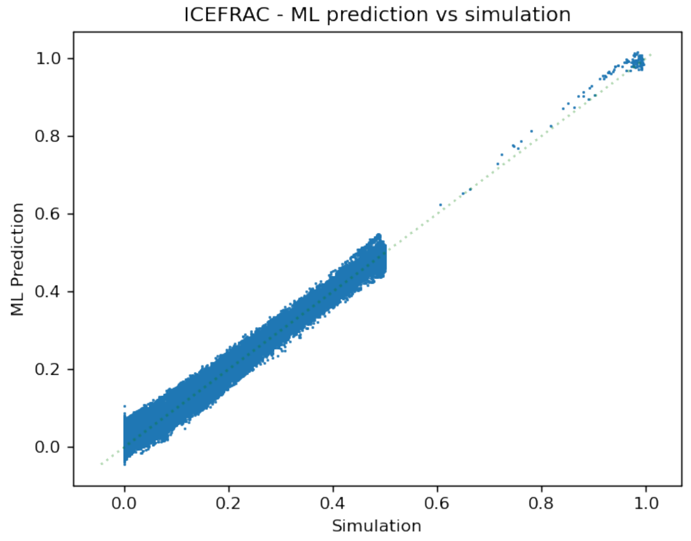
LAYER TESTS – RELATIVE MAX ABSOLUTE ERROR

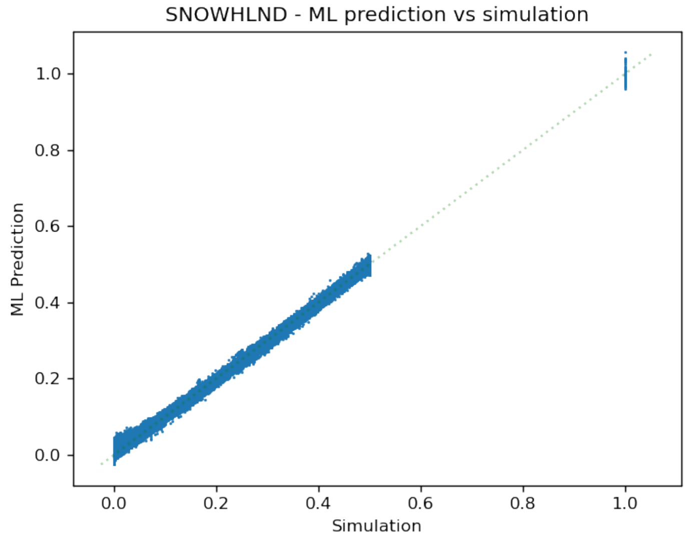


RESULTS

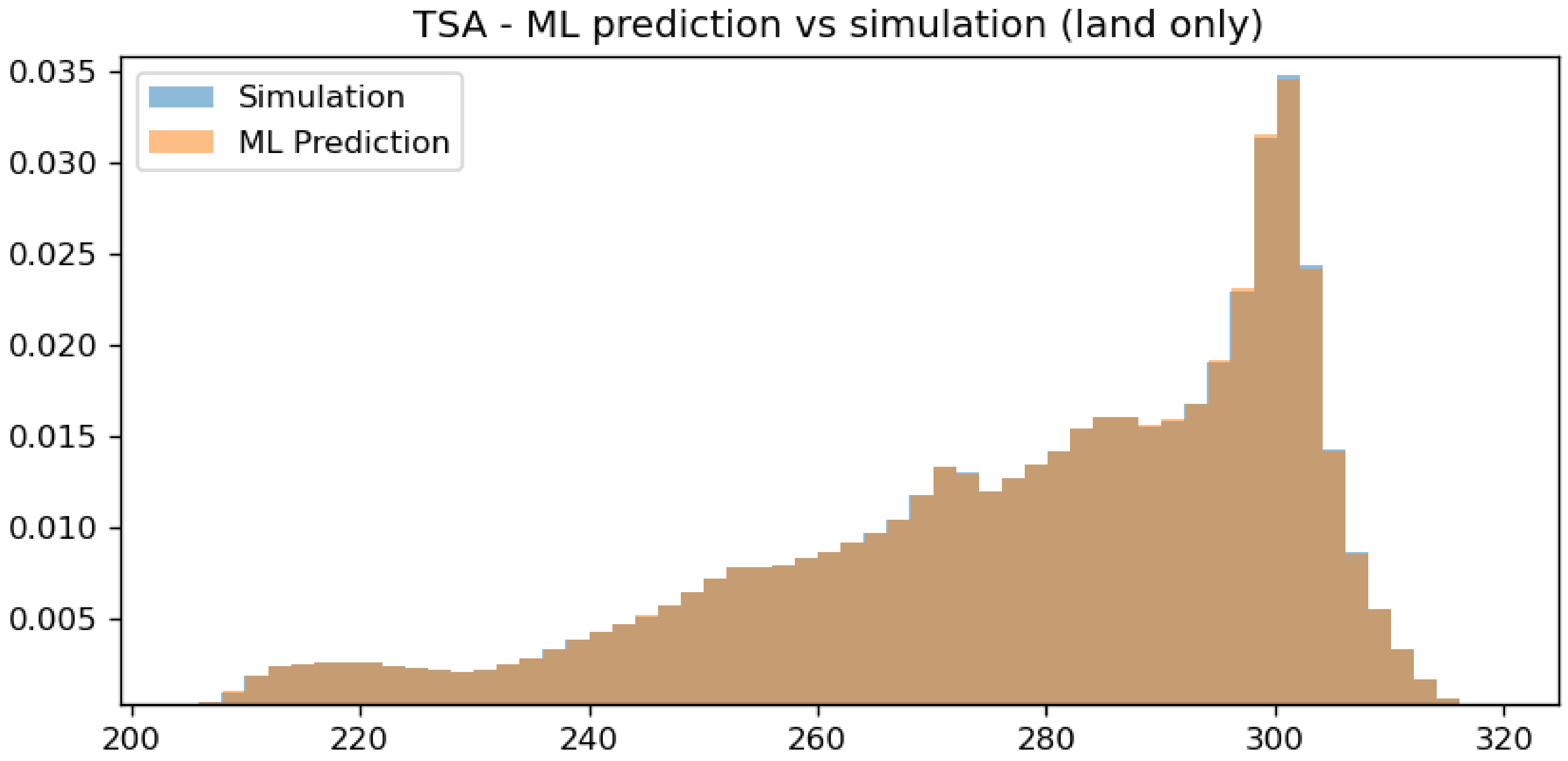

TSA - ML prediction vs simulation
ML Prediction
Simulation
150
200
250
300


ALTMAX - ML prediction vs simulation
ML Prediction
Simulation
17.5
20.0
22.5
25.0
27.5
30.0
32.5
35.0


ICEFRAC - ML prediction vs simulation
ML Prediction
Simulation
0.0
0.2
0.4
0.6
0.8
1.0

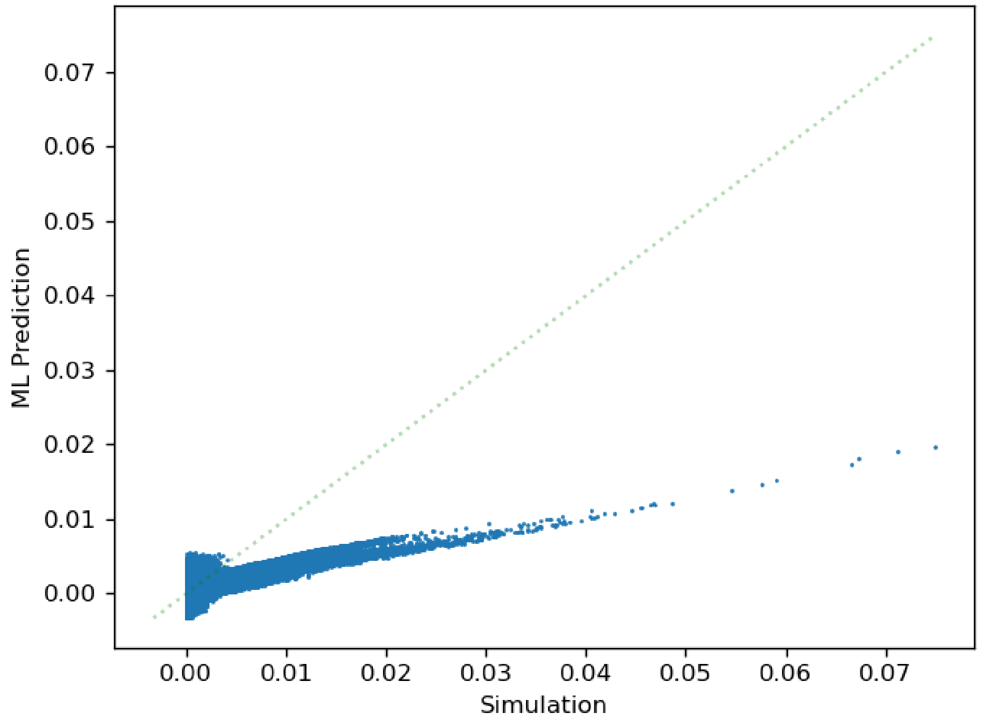

SNOWHLND - ML prediction vs simulation
ML Prediction
Simulation
0.0
0.2
0.4
0.6
0.8
1.0

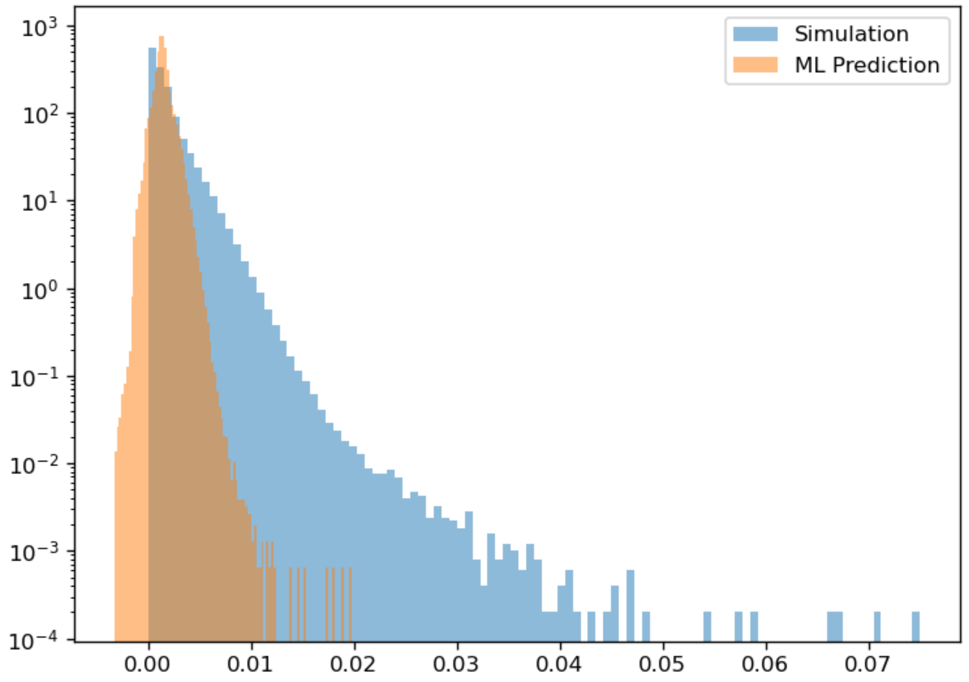
PREDICTION VS SIMULATION - DISTRIBUTION



NOT EVERYTHING IS ROSES

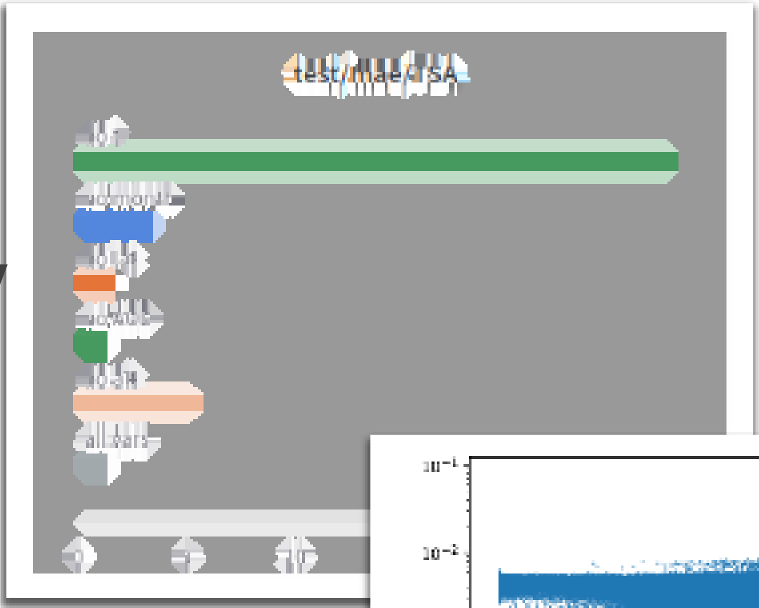
Precipitation

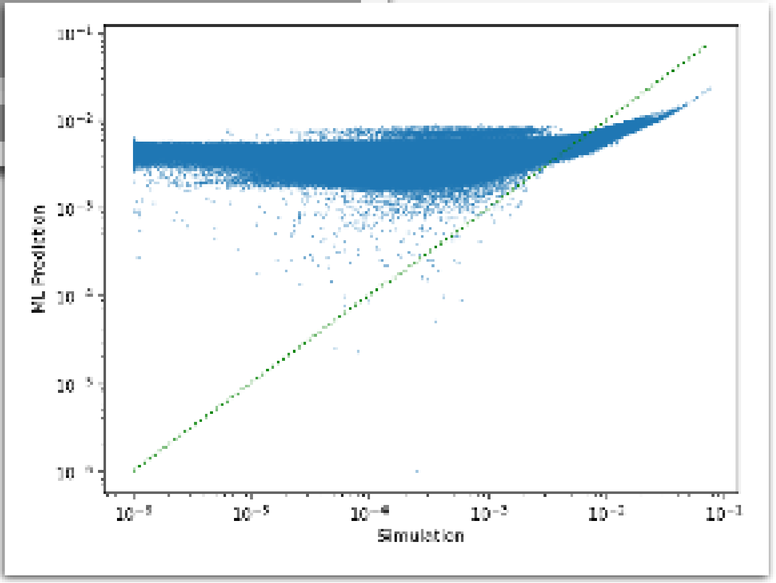




FUTURE

- **Incorporate feedback runs (increased AOD from SAI)**
- **Reduce memorization required of model by adding additional variables (e.g. CO2)**
- **Test autoregressive runs for proper output distribution**
- **Resolve precipitation prediction (think it is** the scale – 5 orders of magnitude!)





QUESTIONS?

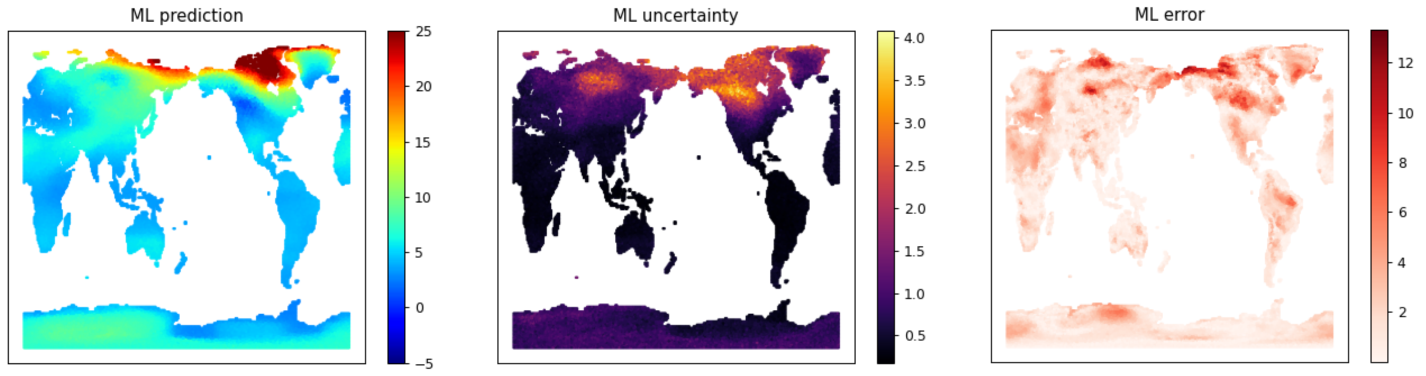
Kevin Potter kmpotte@sandia.gov (ML)

Lauren Wheeler lwheele@sandia.gov (Climate Intervention)

DROPOUT-BASED FCNN ENABLES UNCERTAINTY QUANTIFICATION

Added dropout layers to the neural network (NN)

- Normal use - reduces overfitting
- Introduces stochasticity, forming an ensemble of predictions



(Left) NN prediction for the difference in TSA between a control and feedback run for one month (January 2080). (Center) Uncalibrated uncertainty estimates for NN prediction. (Right) NN error with respect to GLENS simulation data. The uncertainty qualitatively appears to align with high error areas.

Technical details:

- Dropout layers deactivate subsets of NN nodes at random (with probability 0.3 in our implementation).
- During training, this is a typical regularization technique.
- Keeping dropout layers active during inference approximates a Gaussian process [1].
- We make 48 predictions for each input datapoint, generating an ensemble of predictions.
- The standard deviation over the predictions serves an estimate of uncertainty.

[1] Gal, Y., & Ghahramani, Z. (2016, June). Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning* (pp. 1050-1059). PMLR.