

Introduction

Understanding the impacts of climate change is crucial, as climate impacts everyone on this planet. However, climate models like E3SM are computationally expensive to calibrate. We want to create ML surrogate for climate models which are less computationally expensive so that we can use them to reduce the time and resources needed to calibrate climate model.

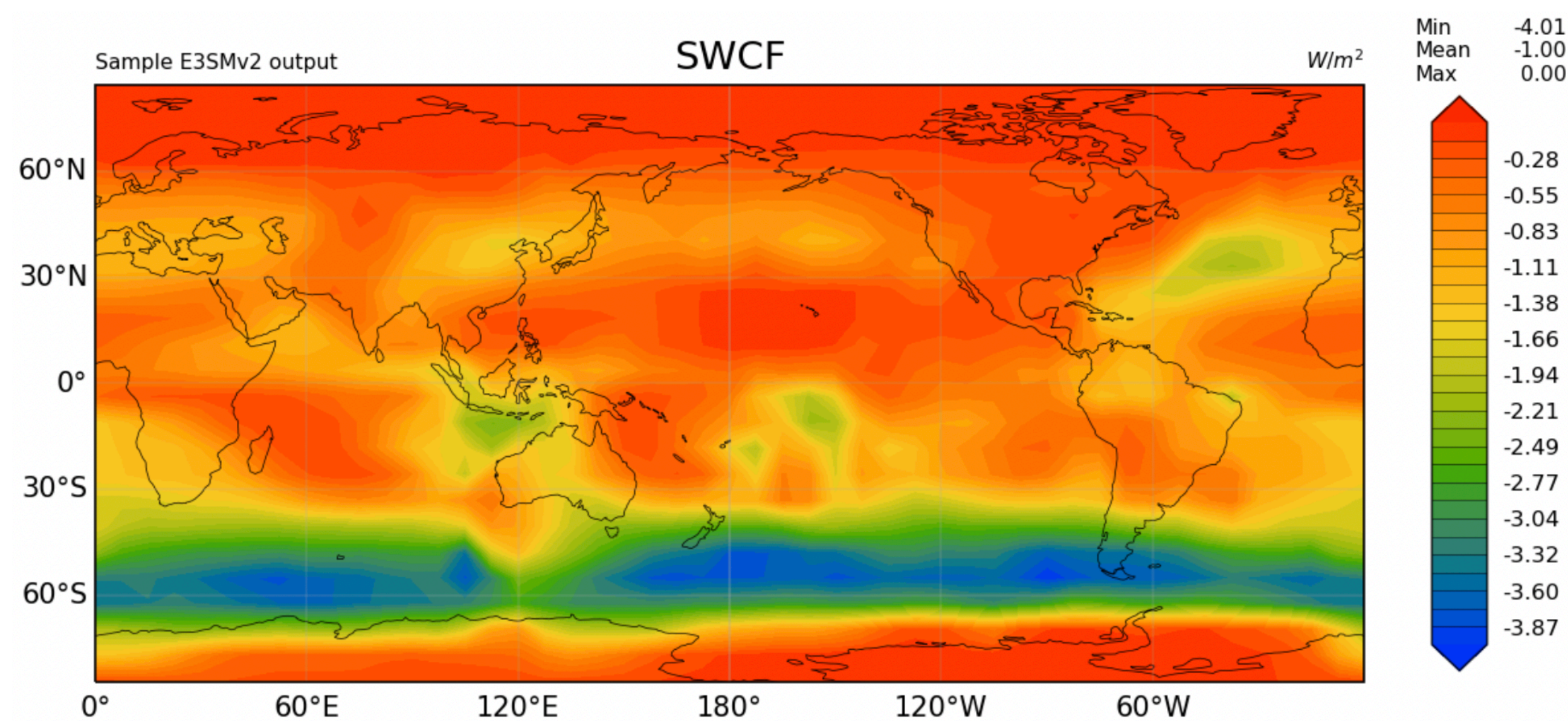


Figure 1. Short-Wave Cloud Forcing Aournd the Globe Predicted by E3SM Model

As we can see a sample output of the E3SM model is a 24 by 48 heat map.

Formulation of the ML Task

- Essentially we are learning a mapping from $X_i \in \mathbb{R}^5$ to $Y_i \in \mathbb{R}^{24 \times 48}$. Hence we have a multi-target regression problem.
- X is 5 parameters of the climate model that we want to create surrogate for
- Y is the 2-d output from the climate model
- we have 250 data points (X, Y) .
- we will use R^2 and MSE as performance measure.

Data Preprocessing

- Data are 250 outputs generated by E3SM model Provided by Sandia National Lab.
- We will normalize X and Y.
- the whole dataset will be partitioned into test(30%) and training set(70%).
- Y will be flattened for models that can only deal with 1-d vector output.
- The flattened Y will be a vector in $\mathbb{R}^{250 \times 1152}$

Implementation and Experimentation

- Models are be implemented and tested with Python packages like sklearn and Tensorflow.
- All models are evaluated hyperparameter-tuned with a 5-fold cross validation
- Deep Models are tuned with Keras-tuner.

Non-Deep Models

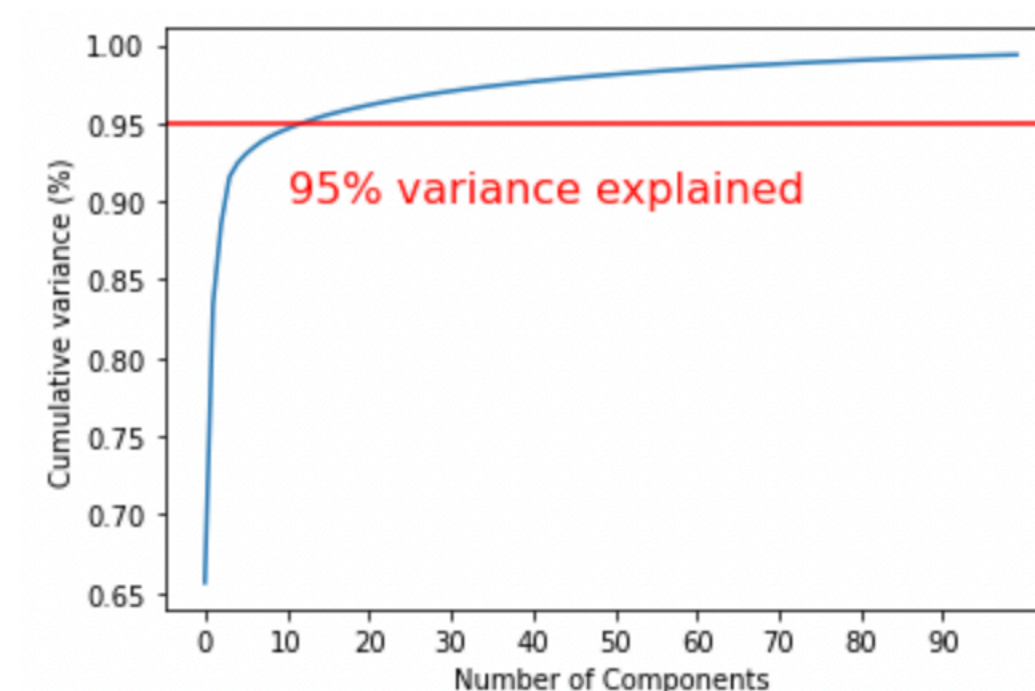
We choose to experiment with multiple models including deep models like fully connected neural network and convolutional neural network, and also tradition regression model like linear regression, etc. Ensemble techniques are also used. Below is the list of non-deep models we have experimented with.

	K Nearest Neighbor	Linear Regression	Lasso	Ridge	Decision Tree	Bagging	Random Forest
R^2	0.15	0.53	0.38	0.5	0.25	0.54	0.48
MSE	0.012	0.006	0.01	0.009	0.008	0.006	0.004

Table 1. Respective R^2 and MSE of each model.

Fully Connected Neural Network With PCA

- The dimension of flattened Y_i is very large: 1152.
- If we want to use models like neural network, it is a good idea for us to reduce the dimension of Y in order to reduce the number of trainable parameters in the model.
- With PCA we can project Y onto a lower dimensional space while preserving as much information as we want.

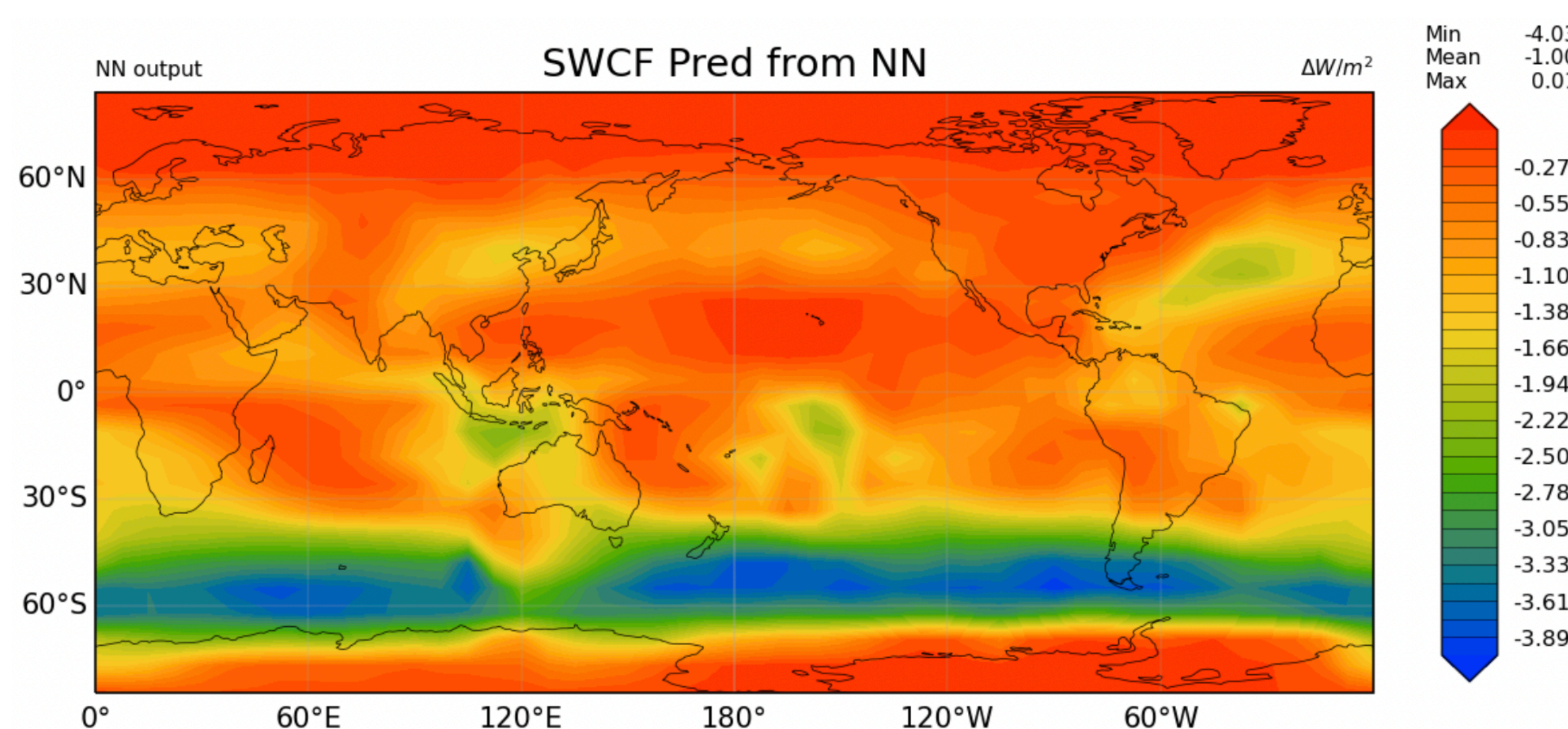


Based on graph above, we choose to reduce Y_i 's dimension from 1152 to 13.

input	layer 1	activation 1	layer 2	activation 2	output	loss
5	96	relu	384	relu	13	MSE

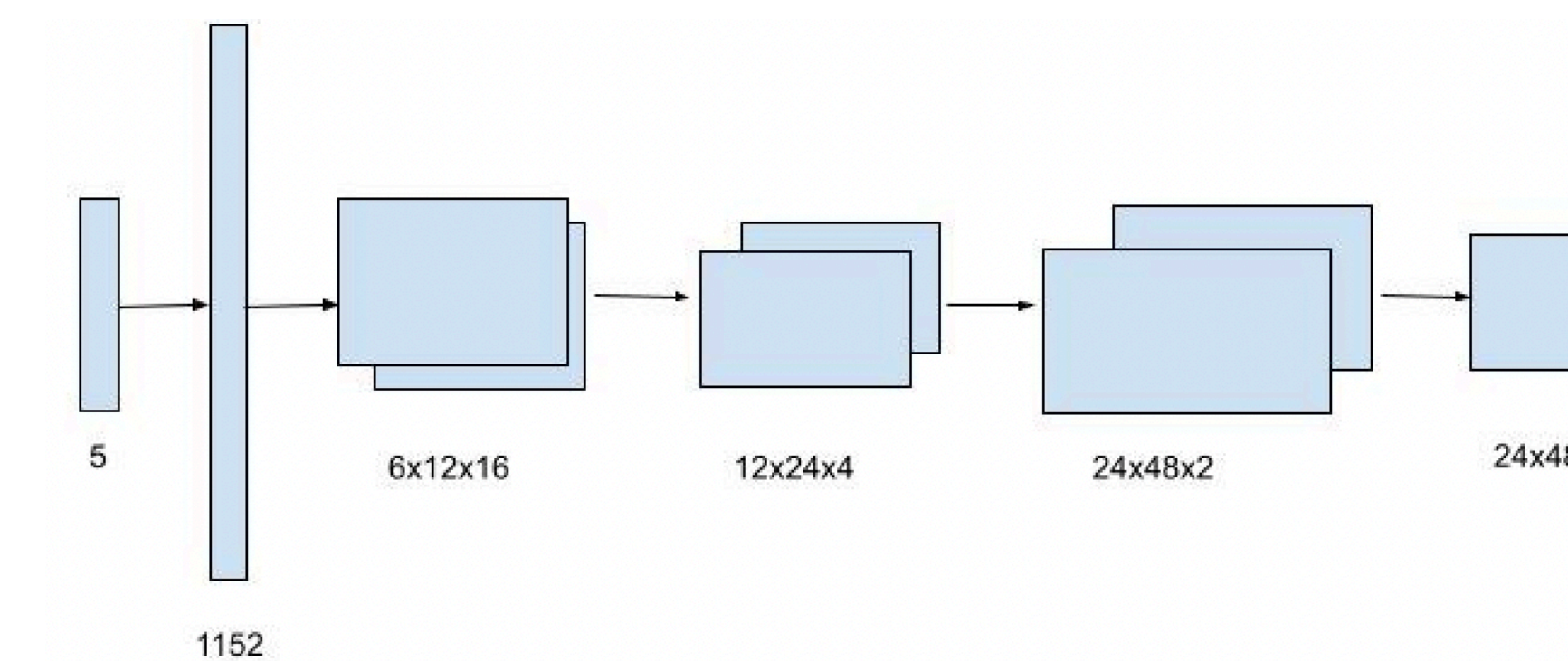
Table 2. Neural Network Architecture

The above fully connected neural network achieves a MSE of 0.002. Below is an visualization of the prediction from the fully connected neural network.



Convolutional Neural Network

The output Y is 2d, so it is natrual to try convolutional neural network which can deal with 2d data. We experimented with the following CNN architecture.



Note here the convolution layers actually up-sample the input using padding. The convolution neural network is able to achieve a similar MSE of 0.002 comparing to the fully connected neural network with only 7585 trainable parameters, which is 80% less than the number of trainable parameters in a fully connecte neural network.

Bayesian Calibration

Getting the Posterior of x_{obs}

Assume that $x_{obs} \sim Uniform(-1, 1)$ and $P(y_{obs}|\theta^*, x_{obs}) \sim N(\mu, \Sigma)$, where $\mu = y_{obs} - f_{surrogate}$ and $\Sigma = \sigma^2 I$, then the posterior of x_{obs} is:

$$p(x_{obs}|y_{obs}, \theta^*) = \frac{p(y_{obs}|\theta^*, x_{obs})p(x_{obs})}{p(y_{obs}|\theta^*)} \propto \frac{1}{(2\pi)^{n/2}\sigma^{2n}} \exp\left(\frac{-1}{2\sigma^2} \sum_{i=1}^n (y_{obs}^{(i)} - f(\theta^*, x_{obs}^{(i)}))^2\right) * p(x_{obs}) \quad (1)$$

Markov Chain Monte Carlo

Now, we sample from the posterior distribution that we have calculated by using HMC(NUTS)

Initialize:

Choose a random x_0 from the sample space $[-1, 1]$

Iterate until convergence:

- Randomly sample $\delta \sim N(0, C')$
- Choose a random candidate $x' = x + \delta$ from the random sample close to x_0
- $\alpha = \frac{P(Y_{obs}|x')p(x')}{P(Y_{obs}|x_0)p(x_0)}$
- Choose $Z \sim U[0, 1]$

$$x_{i+1} = \begin{cases} x' & \text{if } \alpha \geq z \\ x_i & \text{if } \alpha \leq z \end{cases} \quad (2)$$

Posterior Prediction

- $x_{obs} \sim p(x_{obs}|y_{obs}, \theta^*)$
- Use the samples from the MCMC and compute the Maximum a posteriori estimation
- Use the MAP of x_{obs} to query the E3SM model and obtain the Qols