Michael’s June Update

Je-Ching “Michael” Liao

University of Michigan | MS in Data Science

Academia Sinica | RCEC

June 2024

Problem Statement

- Predictors:
  - Brightness Temperature Data from spectrometer (35 channels)
  - Surface Temperature, Surface Pressure, IRT, Surface Relative Humidity
  - Binary predictors: Rain, FLGs, FLGTb
- Predictands:
  - Temperature and Vapor Density across 1001 height intervals (0 ~ 10000m)

Data Processing

- Predictors:
  - Tb data from 24GHz was excluded
  - Only data with “Rain” == 0 are selected: sunny days
  - Binary predictors (Rain, FLGs, FLGTb) were excluded
  - Surface Temperature (Ts), Surface Pressure (Ps), Surface Relative Humidity (RHs): replaced by 0m balloon observed data
  - **Number of training samples tripled** by adding Gaussian Noise
    - +N(0,1) & -N(0,0.8)
  - Total: 37 predictors (Standardized)
- Predictands:
  - Samples without missing values -> training set (# of samples: 53***3**=159)
  - Samples with missing values -> validation set (# of samples: 66)

Benchmarks and Initial Trials (Y: Temp)

- From training set, train test split (0.8/0.2)
- All mean squared error calculated using original 1001 dimensions
  - PCs are inversed transformed before calculation

Mean Squared Error		Training	Test	Duration(s)
_lvl2.nc files		29.664	7.406	N/A
Predictands: 1001 dims	MultiOutputRegressor(RandomForestRegressor)	0.244	1.738	269.69
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.015	1.820	437.85
Predictands: First 5 PCs	MultiOutputRegressor(RandomForestRegressor)	0.688	2.115	1.171
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.508	2.045	1.944

Random Forest Regressor

- Several Decision Trees
- Tree: each maximize Information Gain in terms of Entropy
- Random Forest aggregates the tree outputs
- Less overfitting, handles non-linearity well
- MultiOutputRegressor()

eXtreme Gradient Boosting Regressor

- Gradient Boosting
  - Several weak learners: Decision Trees
  - Sequentially assess and **update** the trees ("Gradient Boosted")
  - Better performing trees: larger weights
  - Monitor samples with large errors: larger weights
- XGBoost
  - Regularization
  - Flexibility
  - Robustness
  - multi_strategy = 'multi_output_tree'

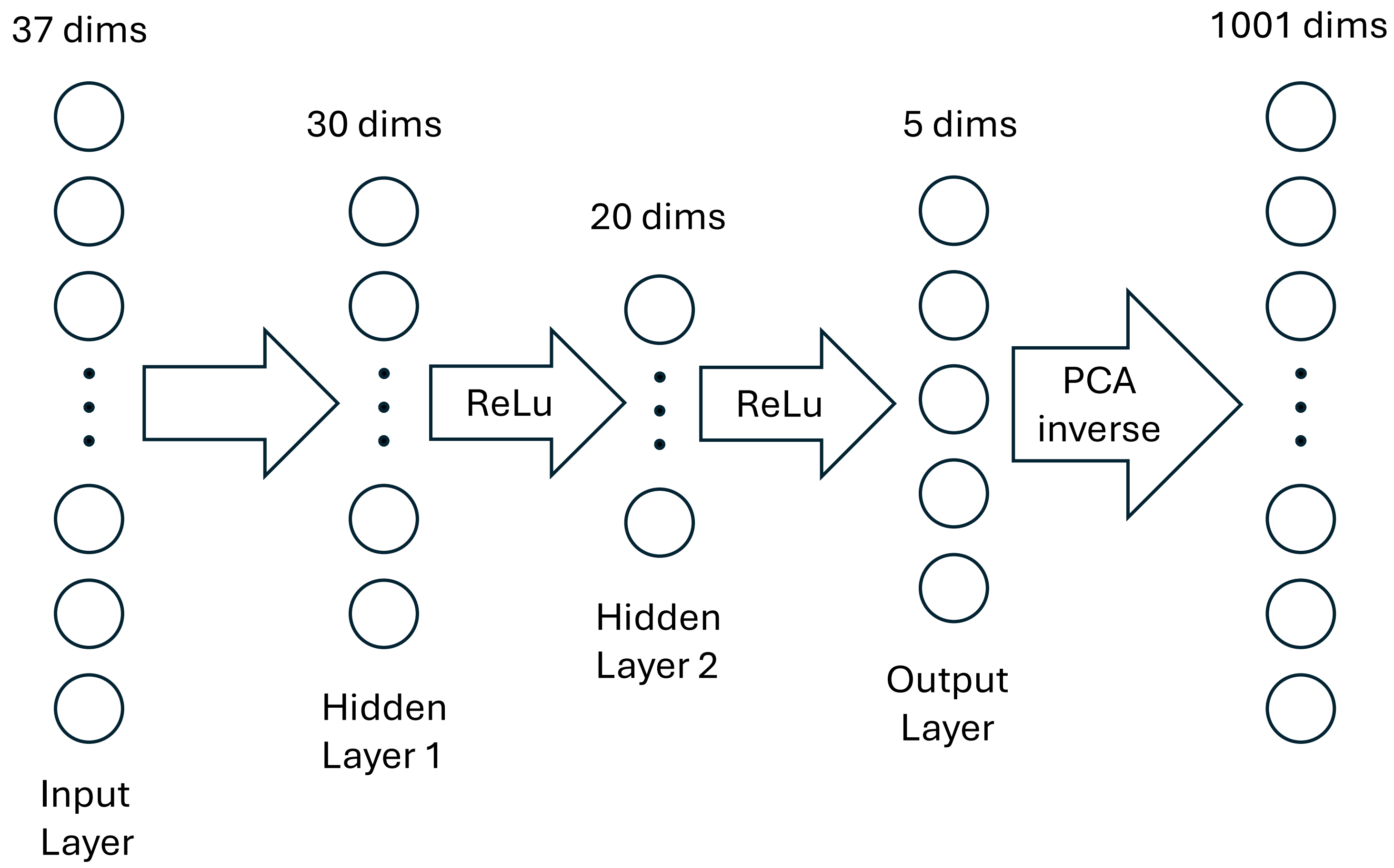
Gaussian Noise Data Augmentation

- Reason:

Mean Squared Error (Both predictands are first 5 PCs)		Training	Test
Before Data Augmentation	MultiOutputRegressor(RandomForestRegressor)	N/A	3.484
	XGBRegressor(*multi_strategy*='multi_output_tree')	N/A	4.448
After Data Augmentation	MultiOutputRegressor(RandomForestRegressor)	0.688	2.115
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.508	2.045

- Number of PCs (5) is also selected with the smallest MSE before data augmentation

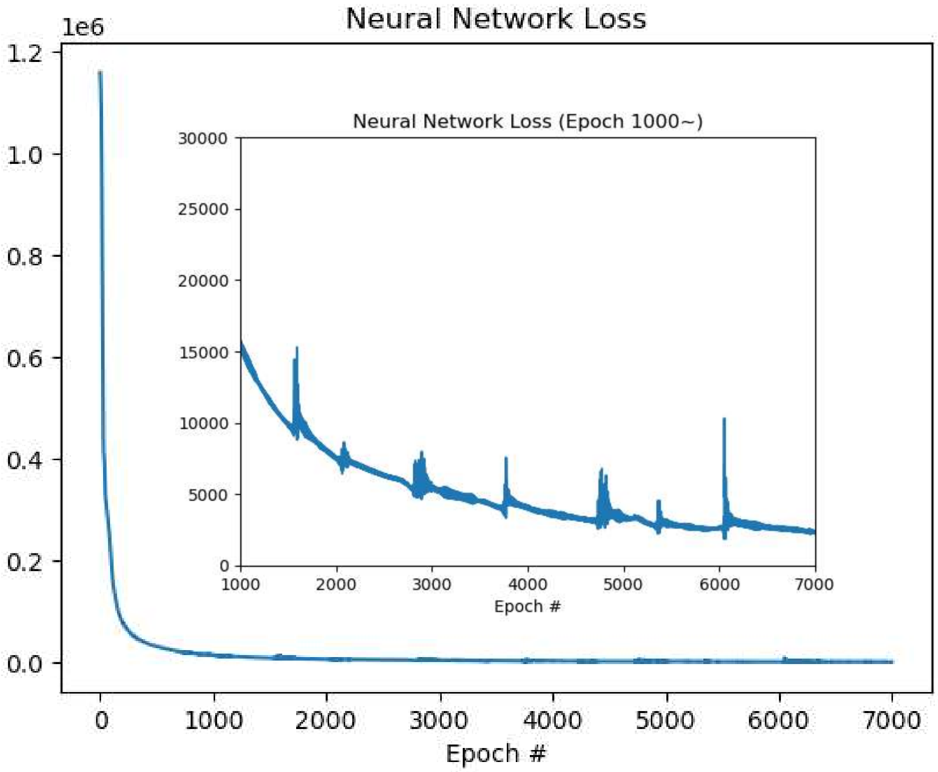
Neural Network Architecture

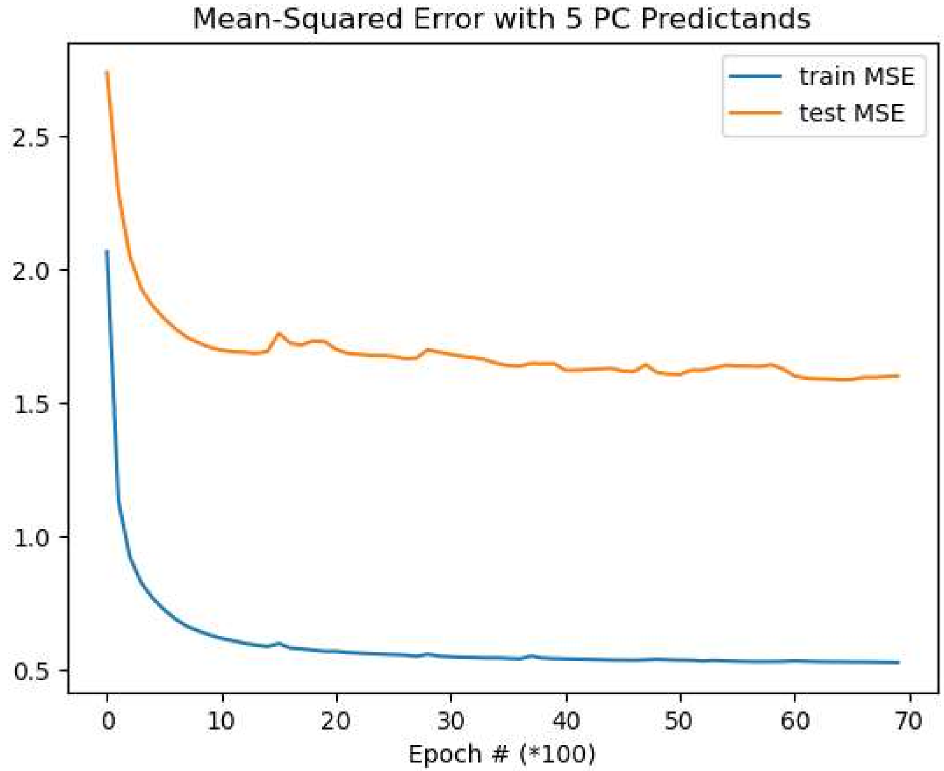


Neural Network Hyperparams

- Criterion: MSELoss(reduction = 'sum')
- Optimizer: AdamW(lr = 0.01)
  - lr: learning rate
- Epochs: 7000
- clip_grad_norm_(max_norm = 5) is applied
  - Clip the norm of gradients: prevent loss explosion

Training Results





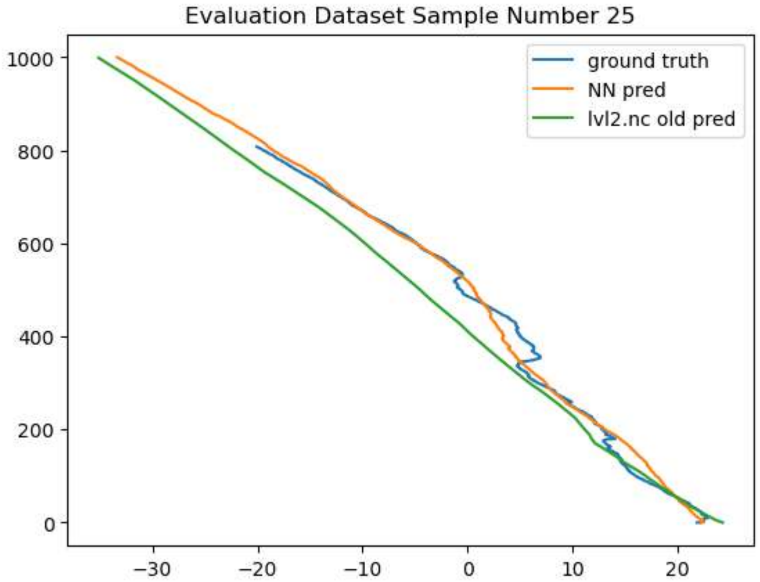
Neural Network Performance

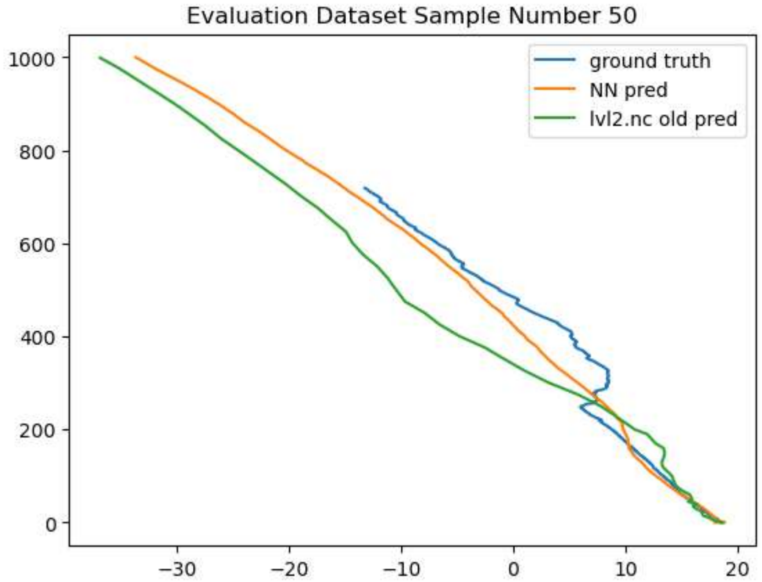
- Neural Network generalize well while having shorter convergence time (No 3 in training, Best in Test, No 3 in Time)

Mean Squared Error		Training	Test	Duration(s)
_lvl2.nc files		29.664	7.406	N/A
Predictands: 1001 dims	MultiOutputRegressor(RandomForestRegressor)	0.244	1.738	269.69
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.015	1.820	437.85
Predictands: First 5 PCs	MultiOutputRegressor(RandomForestRegressor)	0.688	2.115	1.171
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.508	2.045	1.944
	Neural Network(2 Hidden, MSELoss, AdamW)	0.541	1.602	6.706

Evaluation

- Use the dataset with missing values
  - Cannot be used to train and calculate MSE
- Plotting





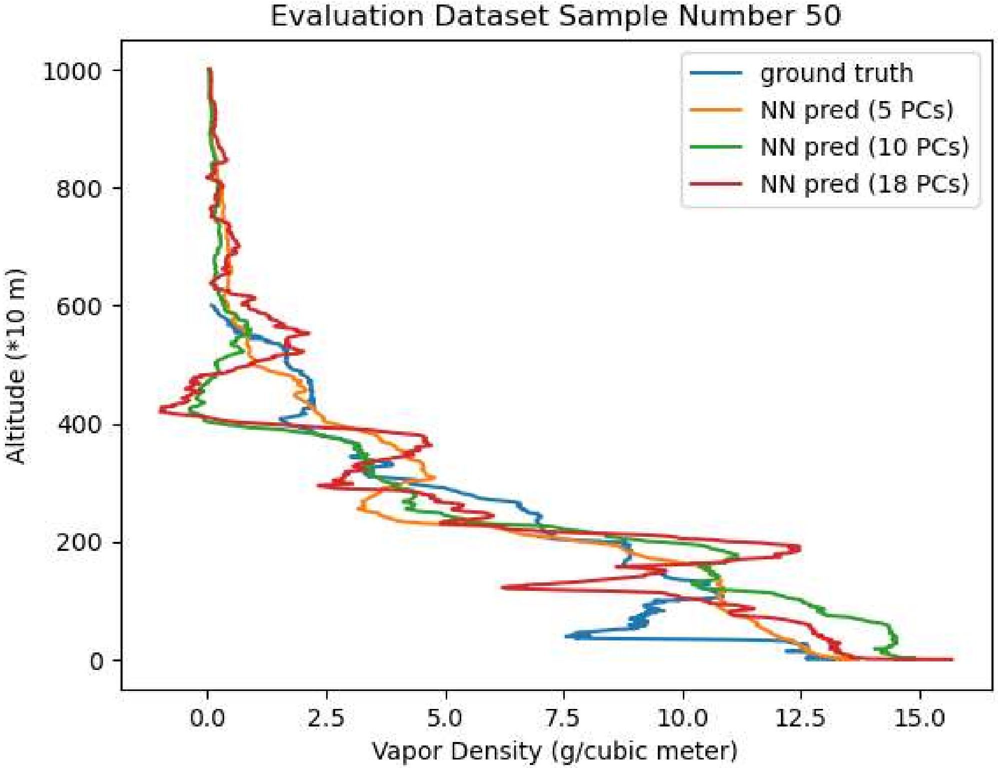
Temperature (T) Predictions

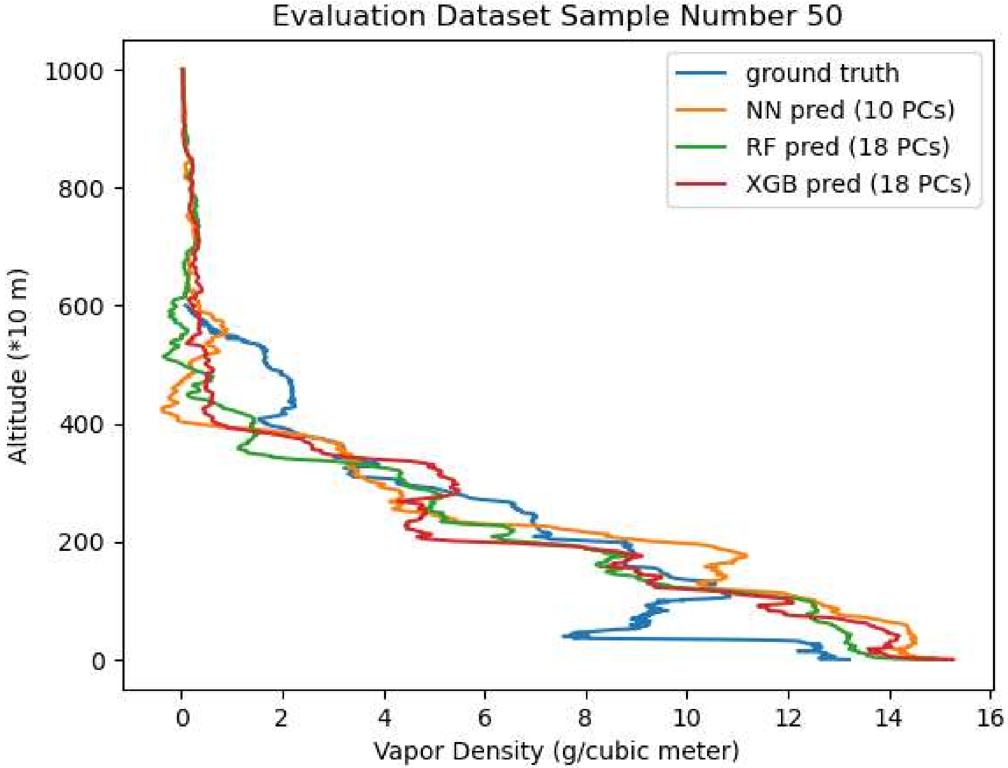
Mean Squared Error (Predictands in different PCs)		Training	Test	Duration(s)
_lvl2.nc files		29.664	7.406	N/A
Original 1001 dims	MultiOutputRegressor(RandomForestRegressor)	0.244	1.738	269.69
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.015	1.820	437.85
First 5 PCs	MultiOutputRegressor(RandomForestRegressor)	0.688	2.115	1.171
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.508	2.045	1.944
	Neural Network (2 Hidden Layers, 7000 iterations)	0.541	1.602	6.706
First 10 PCs	Neural Network (2 Hidden Layers, 7000 iterations)	0.200	1.600	6.953
First 18 PCs	MultiOutputRegressor(RandomForestRegressor)	0.260	1.832	5.195
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.043	2.101	8.358
	Neural Network (2 Hidden Layers, 7000 iterations)	0.157	1.720	6.516
First 27 PCs	Neural Network (2 Hidden Layers, 7000 iterations)	0.137	1.677	7.062

Vapor Density (Qv) Predictions

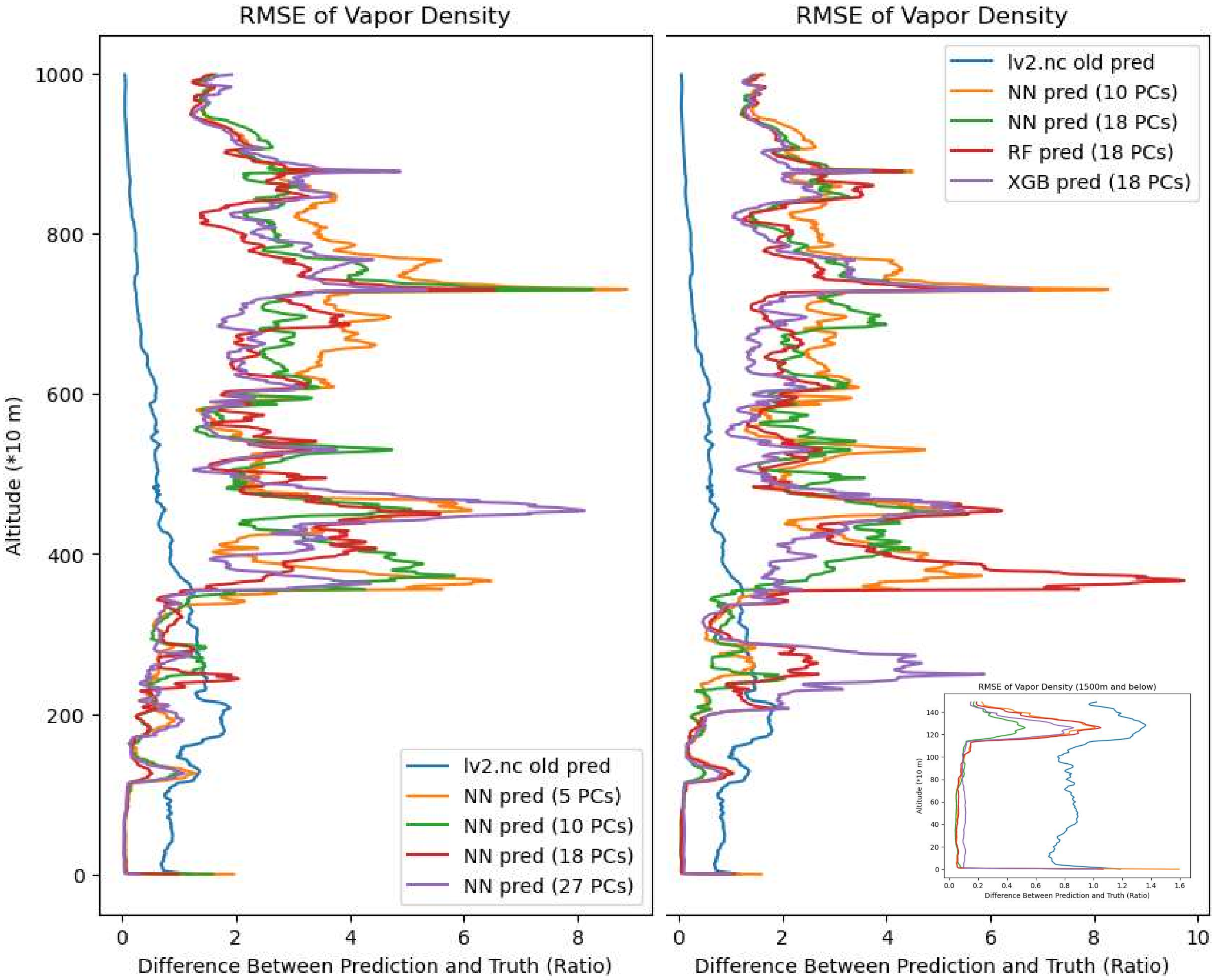
Mean Squared Error (Predictands in different PCs)		Training	Test	Duration(s)
_lvl2.nc files		5.188	0.705	N/A
Original 1001 dims	MultiOutputRegressor(RandomForestRegressor)	0.078	0.596	270.81
	XGBRegressor(*multi_strategy*='multi_output_tree')	8.87e-6	0.569	465.56
First 5 PCs	MultiOutputRegressor(RandomForestRegressor)	0.318	0.701	1.337
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.269	0.727	2.380
	Neural Network (2 Hidden Layers, 7000 iterations)	0.275	0.737	6.973
First 10 PCs	Neural Network (2 Hidden Layers, 7000 iterations)	0.096	0.674	6.953
First 18 PCs	MultiOutputRegressor(RandomForestRegressor)	0.097	0.565	5.122
	XGBRegressor(*multi_strategy*='multi_output_tree')	0.029	0.639	8.358
	Neural Network (2 Hidden Layers, 7000 iterations)	0.070	0.577	6.516
First 27 PCs	Neural Network (2 Hidden Layers, 7000 iterations)	0.058	0.565	6.608

Evaluation: Single Sample Comparison

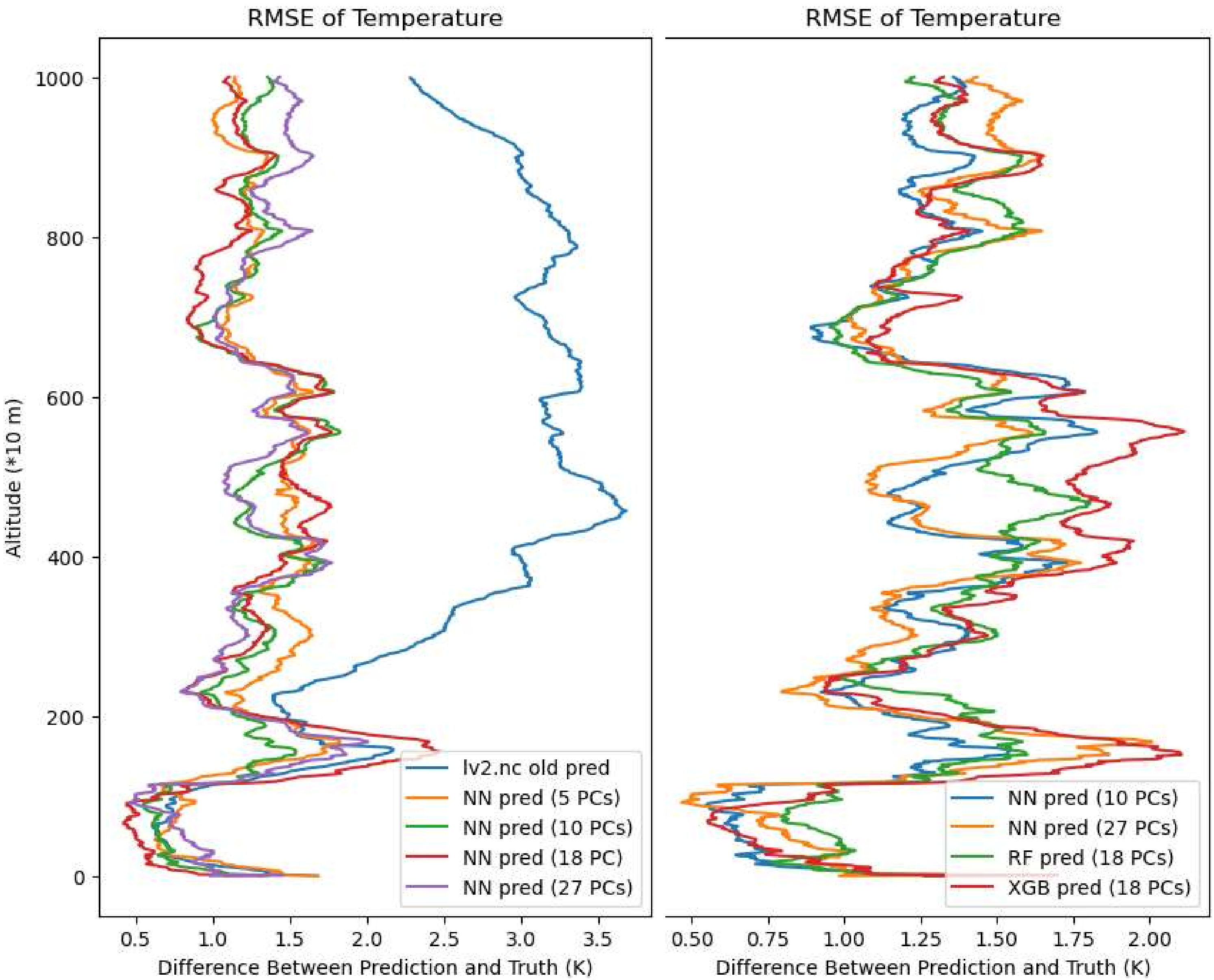




Evaluation Qv RMSE against Altitude



Evaluation Temp RMSE against Altitude



Model Selection + Relative Humidity Retrieval

- Temperature: Neural Network with 10 PCs
- Vapor Density: Neural Network with 18 PCs

Then, saturated vapor pressure (e_s) and the actual vapor pressure (e) can be calculated using the formula listed below:

$$e = 6.11 \times 10^{\left(\frac{7.5 \times T_d}{237.3 + T_d}\right)} \qquad e_s = 6.11 \times 10^{\left(\frac{7.5 \times T}{237.3 + T}\right)}$$

For a bonus answer, after calculating both vapor pressures the relative humidity (rh) can be calculated using the equation below:

$$rh = \frac{e}{e_s} \times 100$$

$$e = \rho_v R_v T$$

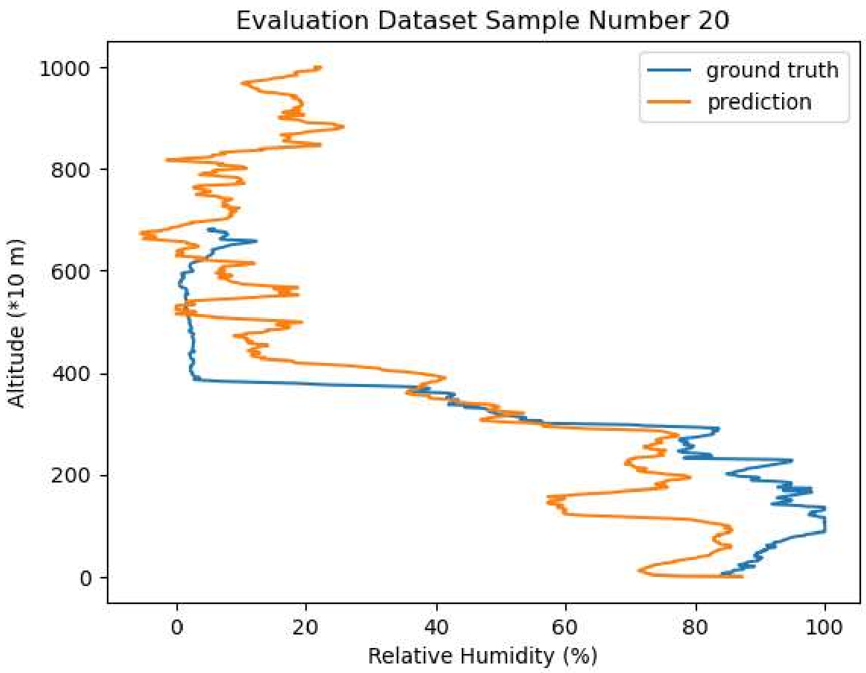
e: $in\ Pa$

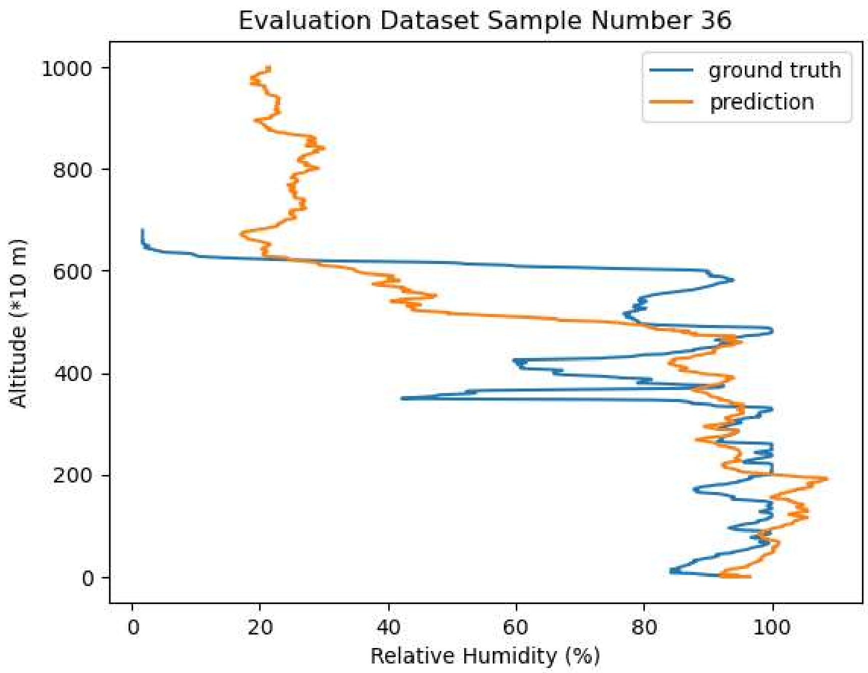
ρ_v: $vapor\ density\ in\ kg/m^3$

T: $temperature\ in\ K$

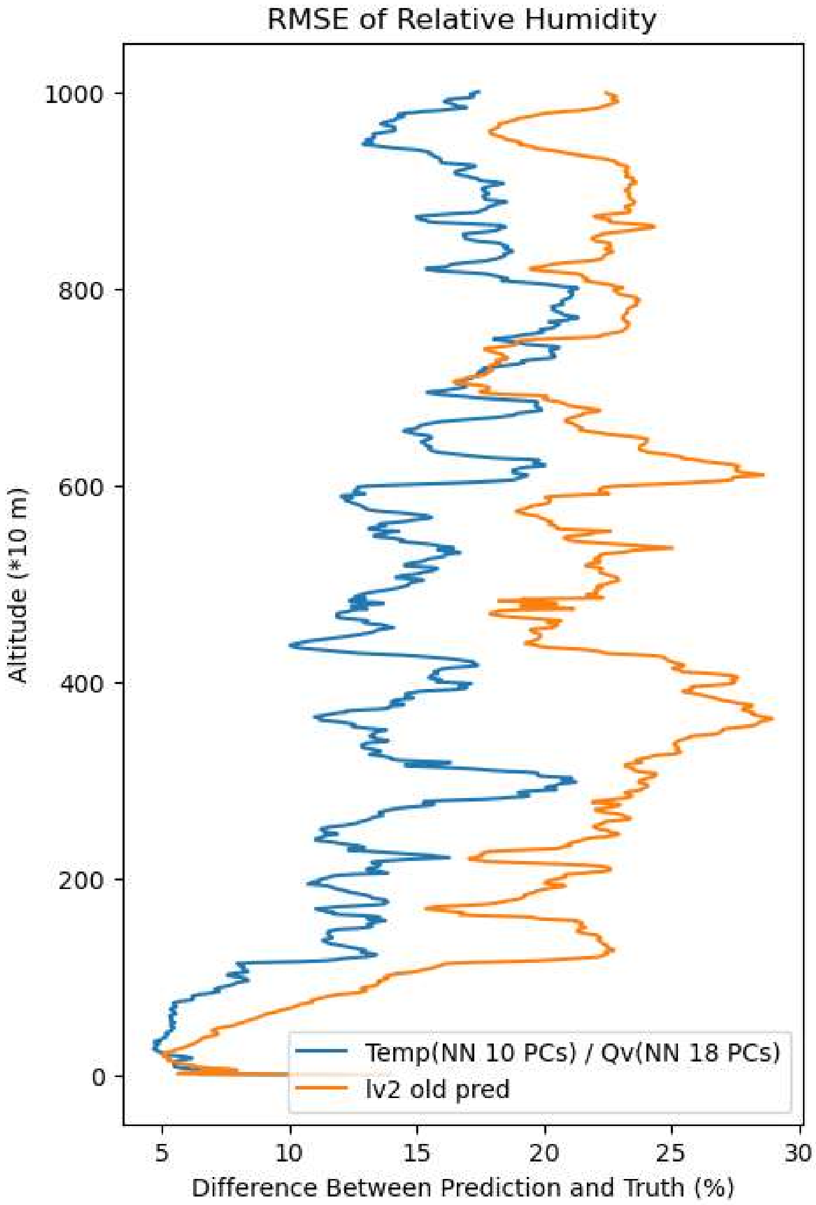
$Rv = 461.5$

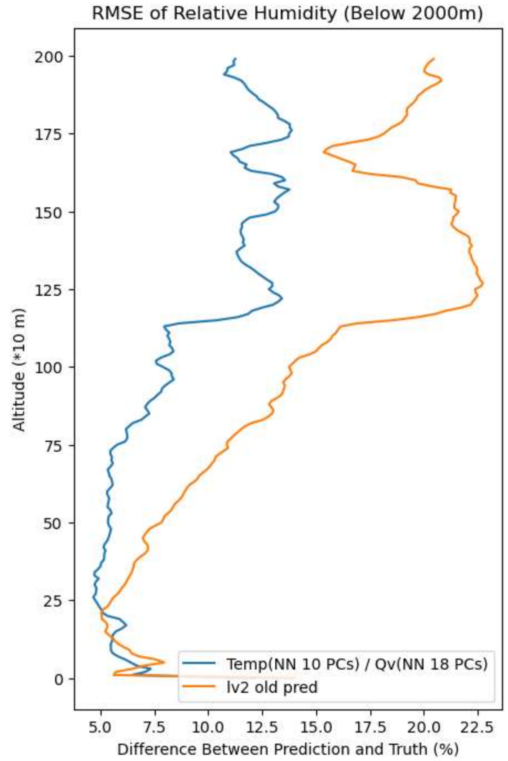
Relative Humidity Retrieval





Evaluation: Relative Humidity RMSE against Altitude





Future Steps

- Open to suggestions