An Introduction to Calibration Function Development

Georgi Tancev

January 30, 2020

Case Example: Introduction

- Sensor generates voltage due to reaction(s) with chemical compounds.
- Voltage is proportional to amount of reactants.
- Mapping sensor signal to gas reference concentration ($f : S \to R$), but accounting for cross-sensitivities and other phenomena.

Case Example: Conditions

- ▶ The aim is to gather prior information from published data.
- ▶ Experimental data from DeVito (2009): Measurement campaign took place at a main street in the center of an Italian city characterized by heavy car traffic.
- ▶ Raw sensor data of CO_2, CO, NO_x, NO_2, O_3, temperature, and absolute/relative humidity.
- ▶ Reference data of NO_x, NO_2, CO, C_6H_6.
- ▶ Each sensor/reference is stationary without replicates and a sampling rate of $1\ h^{-1}$ for a period of one year.
- ▶ Is it possible to map raw sensor data to reference data?

Case Example: Constraints

- ▶ Model inference should be feasible with respect to time and space complexity in case model and/or calibration are run locally.

- ▶ Backpropagation, which is used for neural network inference, has a time complexity of $\mathcal{O}(n \cdot m \cdot h^k \cdot o \cdot i)$ with n training samples, m features, k hidden layers with h latent variables each, o outputs, and i iterations.

- ▶ Gradient descent has time complexity of $\mathcal{O}(n \cdot m \cdot o \cdot i)$, since a neural network without latent variables and with linear activation is equal to a multi-linear regression.

- ▶ Stochastic gradient descent has even lower time complexity due to smaller batch size n.

- ▶ In addition, space complexity has to be considered as some algorithms are memory-based and need to store all data.

- ▶ Ideally, model M_{t-1} from previous calibration time point $t-1$ is retrained by performing several iterations of (stochastic) gradient descent starting from parameters W_{t-1}. This ensures that the new model M_t with parameters W_t does not change too much from its parameter distribution $p(W)$.

Case Example: Workflow

1. Select Scikit-Learn and Python as tools for model development.
2. Start with a complex model and try to predict reference concentrations from sensor voltages.
3. Tune hyperparameters to obtain a good benchmark.
4. Prune model by keeping loss approximately constant.
5. Inspect model to understand its behavior.

Methods: Preprocessing

- Drift is unknown, assuming first six months as free from drift and discarding the rest.
- Standard scaling to zero mean and unit variance.
- Dropping missing values instead of imputing.
- Removing outliers with isolation forest.
- Shuffling.
- Final dataset consists of $N = 2'700$ samples over six months.
- Splitting data in $\frac{7}{10}N$ training and $\frac{3}{10}N$ test set.

Methods: Model Development

- ▶ Choosing a neural network (multi-layer perceptron) with rectifier activation function as model due to its capability to learn non-linearities.
- ▶ Performing a randomized grid search to optimize bias/variance by minimizing mean squared error via 5-fold cross-validation with respect to
    - ▶ L_2-regularization parameter $\alpha \in \{10^{-2+0.08j} \mid j \in \mathbb{N}, 0 \leq j \leq 50\}$,
    - ▶ number of hidden layers $k \in \{1, 2, 3\}$,
    - ▶ and number of latent variables $h \in \{5, 10, 15\}$.
- ▶ This leads to a deterministic machine (maximum likelihood estimate), although
    - ▶ data is noisy (aleotoric uncertainty)
    - ▶ and sensors come from a population (epistemic uncertainty).
- ▶ Pruning model by removing layers and latent variables unless loss changes significantly from benchmark.

Results and Discussion

▶ Final model consists of 1 layer with 10 latent variables and regularization parameter $\alpha = 1.6$
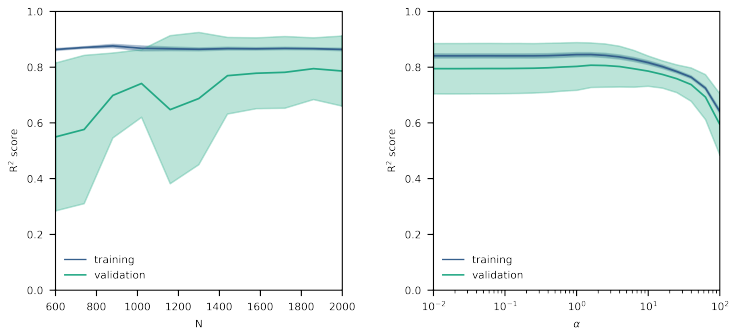
▶ Amount of data seems reasonable.

Figure: Training and validation curves for final model.

Results and Discussion

- R^2 score is 0.83, β is 0.81, and mean squared error is 0.1 averaged over all references on test data, which is good compared to literature.
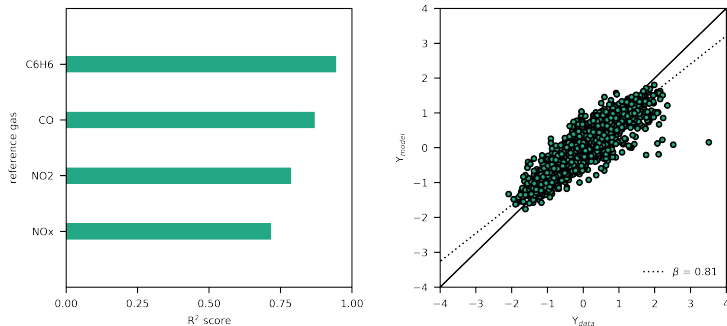- C_6H_6 has no reference but shows best prediction performance.

Figure: Model performance with respect to individual references.

Results and Discussion

- ▶ Only two sensors are important and rest is apparently redundant.
- ▶ The instruments of low importance could suffer from correlations since permutations can be compensated by correlated features.
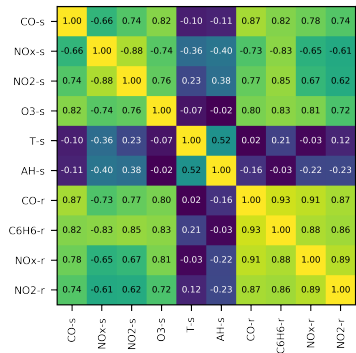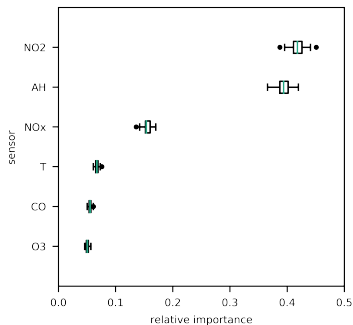

Figure: Importance of individual sensors and Spearman rank correlation of signals.

Results and Discussion

- One sensor detects and measures several gases in cases of highly correlated signals and/or references.
- Performing laboratory experiments with varying conditions helps decorrelating signals and removing sampling bias.

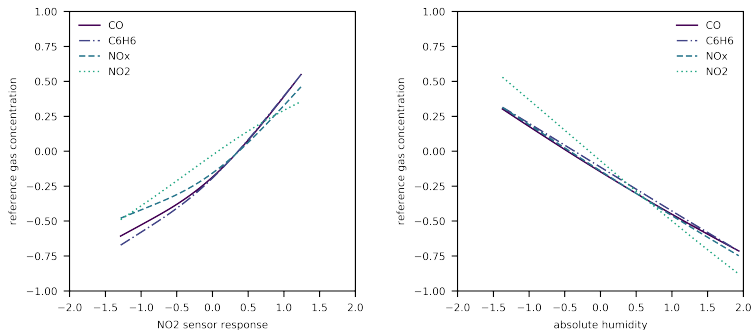

Figure: Partial dependence plots of references on sensors.

Results and Discussion

- ▶ Non-linearities are not very pronounced in this particular case.
- ▶ Non-linearities can also be modeled using parametric linear models by performing some feature engineering, e.g. basis expansion, or non-parametric random forests with deep trees.
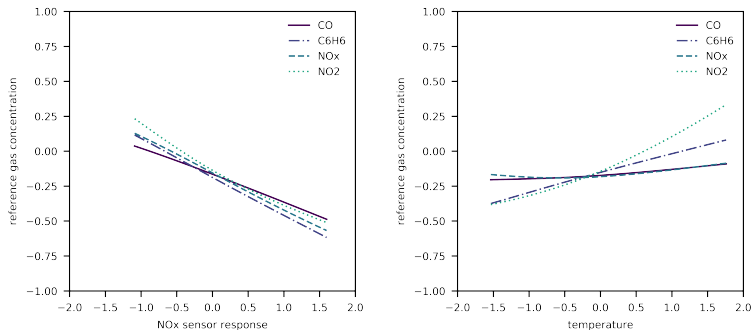

Figure: Partial dependence plots of references on sensors.

Results and Discussion

▶ Underlying pollutant generation processes are not independent, and hence, chemical concentrations are conditional dependent.
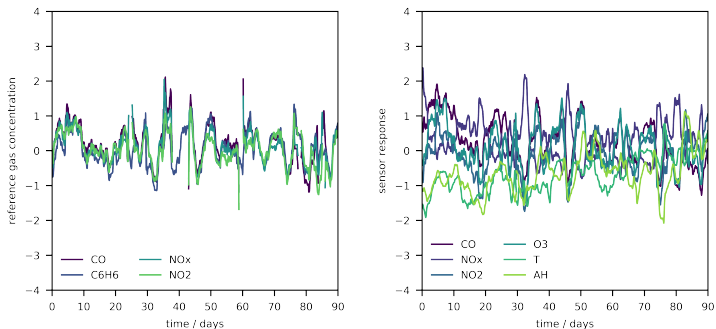

Figure: Time series of references and sensor responses.

Conclusion

- Inspecting model is a key aspect in machine learning.
- Examining data distribution and quality is of high importance.
- Uniform sampling under laboratory conditions with some design of experiments as complementary data acquisition method is unavoidable for a good model.
- Choice of model class is less likely to be relevant with respect to performance, which would be consistent with the ambiguity in the literature.

Appendix

Deterministic Modeling with TensorFlow

- Same architecture but optimized split in TensorFlow with early stopping instead of regularization.
- R^2 score is 0.86, β is 0.84, and mean squared error is 0.1 averaged over all references on test data, which is consistent with previous result.
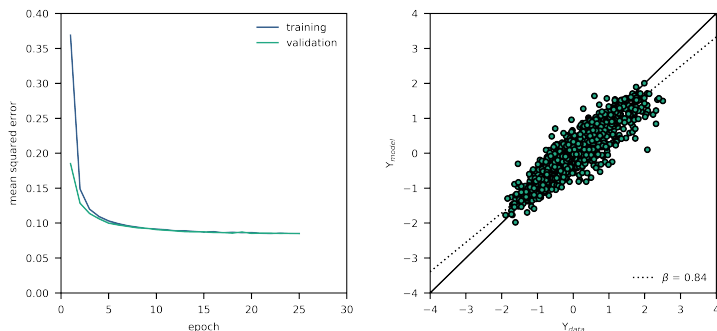

Figure: Model summary with loss and metrics.

Probabilistic Modeling with TensorFlow Probability

- ▶ Modeling of aleotoric uncertainty (noise) by Monte-Carlo experiments.
- ▶ To obtain a prediction y_i for measurement x_i, k samples from posterior distribution $p(y|x_i)$ are drawn and expectation $y_i = \mu_{y|x_i}$ is calculated.
- ▶ Each prediction y_i has information about uncertainty due the standard deviation $\sigma_{y|x_i}$ from the k draws.
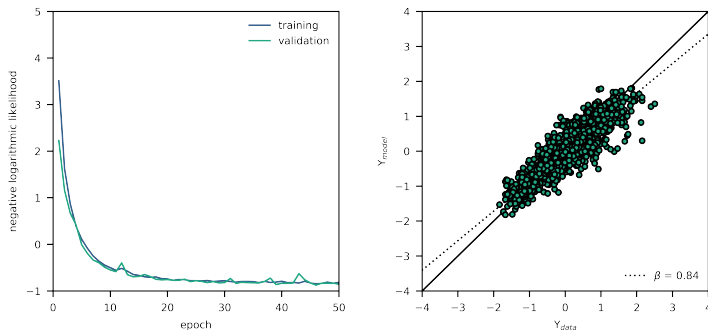

Figure: Model summary with loss and metrics.

Probabilistic Modeling with TensorFlow Probability

▶ Case example with synthetic time series data from three sensors and three references.

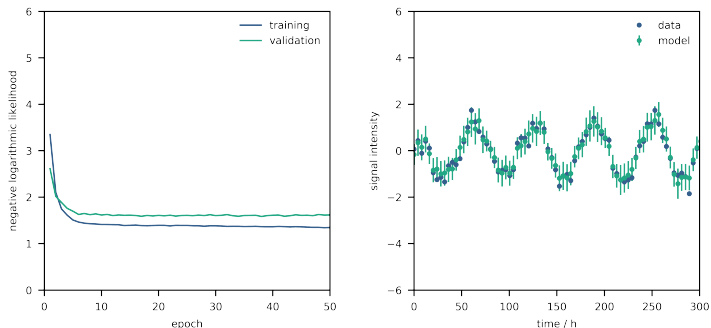▶ Prediction is an aggregated result of $\mu_{y|x_i} \pm 1.96 \cdot \sigma_{y|x_i}$ (true value lies in this interval with a probability of 95%).

Figure: Training curve with sample prediction.