Machine learning for estimating parameters of a convective-scale model: A comparison of neural networks and random forests

Stefanie Legler[1], Tijana Janjic[1], Mohammad Hossein Shaker[2], Eyke Hüllermeier[2]

[1]Mathematical Institute for Machine Learning and Data Science
KU Eichstätt-Ingolstadt
E-Mail: stefanie.legler@ku.de, tijana.janjic@ku.de

[2]Institute of Informatics
University of Munich (LMU)
Munich Center for Machine Learning (MCML)
E-Mail: mhshaker@mail.upb.de, eyke@lmu.de

1 Introduction

Errors and inaccuracies in the representation of clouds in convection-permitting numerical weather prediction models can be caused by various sources, including the forcing and boundary conditions, the representation of orography, and the accuracy of the numerical schemes determining the evolution of humidity and temperature. Moreover, the parametrization of microphysics and the parametrization of processes in the surface and boundary layers do have a significant influence. These schemes typically contain several tunable parameters that are either non-physical or only crudely known, leading to model errors and imprecision. Furthermore, not accounting for uncertainties in these parameters might lead to overconfidence in the model during forecasting and data assimilation (DA).

Traditionally, the numerical values of model parameters are chosen by manual model tuning. More objectively, they can be estimated from observations

DOI: 10.58895/ksp/1000151141-1 erschienen in:
Proceedings – 32. Workshop Computational Intelligence: Berlin, 1. - 2. Dezember 2022
DOI: 10.58895/ksp/1000151141 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000151141/

by the so-called augmented state approach during the data assimilation [7]. Alternatively, the problem of estimating model parameters has recently been tackled by means of a hybrid approach combining DA with machine learning, more specifically a Bayesian neural network (BNN) [6]. As a proof of concept, this approach has been applied to a one-dimensional modified shallow-water (MSW) model [8].

Even though the BNN is able to accurately estimate the model parameters and their uncertainties, its high computational cost poses an obstacle to its use in operational settings where the grid sizes of the atmospheric fields are much larger than in the simple MSW model. Because random forests (RF) [2] are typically computationally cheaper while still being able to adequately represent uncertainties, we are interested in comparing RFs and BNNs. To this end, we follow [6] and again consider the problem of estimating the three model parameters of the MSW model as a function of the atmospheric state.

2 Model and methods

2.1 The MSW model

The MSW model is used to generate the true atmospheric state as well as the forecasts. This simple toy model realizes a mapping $x(t+dt) = MSW_\theta(x(t))$ to simulate the development of wind (u), clouds (h) and rain (r), and can be used to study new DA algorithms. A one dimensional grid with 250 grid points is used, yielding a state vector of the form:

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{u}(t) \\ \mathbf{h}(t) \\ \mathbf{r}(t) \end{bmatrix} \in \mathbb{R}^{750}. \tag{1}$$

In a realistic setting we do not have access to the true atmospheric state but only to observations which are sparse and noisy and only available at distinct times. We simulate this by adding noise to the true atmospheric state and only observe every 60 model time steps. Furthermore, we observe all variables of the MSW model only at those grid points where $r > 0.005$ to simulate radar data. The MSW model parameters to be estimated are the rain removal rate α, the constant value for the geopotential ϕ_c, and the threshold for the fluid height h_r. All model parameters are constant in space and yield a three-dimensional parameter vector of the form

$$\theta(t_k) = \begin{bmatrix} \alpha(t_k) \\ \phi_c(t_k) \\ h_r(t_k) \end{bmatrix} \in \mathbb{R}^3. \tag{2}$$

that will be estimated in discrete times t_k. The parameters for training and testing are taken from uniform distributions with the same bounds as in [7, 6] and rescaled to the unit interval [0,1] before training.

2.2 Machine learning

A BNN and a RF regressor are used to estimate the three model parameters of the MSW model as a function of a snapshot in time of the atmospheric state (1). This results in an input size of 750 and an output size of 3. For the BNN, stochastic components are introduced over the weights [5] of a fully connected neural network with three hidden layers (see [6] for details of the architecture). The priors for the weights are normal distributions with mean 0 and standard deviation 1. These were optimized via ELBO-based stochastic variational inference [4] using Pyro [1]. The RF consists of 100 trees with the minimum sample size for a split set to five.

2.3 Data assimilation

In reality it is not possible to produce accurate weather forecasts by using only a dynamical model, but it is necessary to update the forecast at certain time intervals using current observations. This process is also called the DA cycle, and the updated forecast is called the analysis. Panels II. + III. in Fig. 1 outline the usual steps of such a DA cycle: at time t, a weather forecast for time $t+dt$ is generated using the MSW model which starts from the current analysis ensemble: $x^{fc}(t+dt) = MSW_\theta(x^{an}(t))$. Once the time $t+dt$ is reached, and

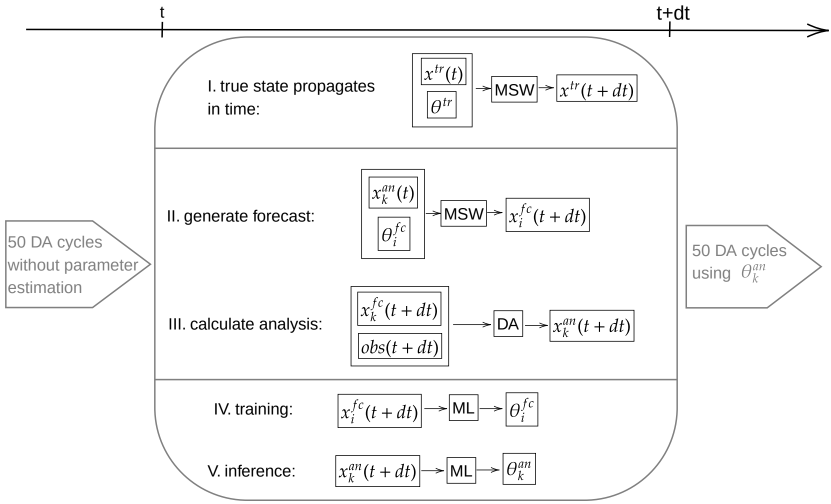

Figure 1: Sketch of the combined DA and ML algorithm to estimate the state and parameters.

thus observations of the true atmosphere are available, $x^{fc}(t+dt)$ is updated with the current observations obs_{t+dt} using a DA algorithm. We utilize a stochastic Ensemble Kalman Filter in this work [3]. For this algorithm a time-varying sample covariance is calculated and used for each forecast ensemble member, resulting in a new analysis ensemble $\{x_k^{an}(t+dt)\}$. This analysis ensemble can then be used to generate the next weather forecast for the time $t+2dt$. Note, if one would simply use $x^{fc}(t+dt)$ to generate the next forecast for $t+2dt$, instead of the analysis, the state error would grow over time and make the forecast inaccurate.

3 Experiments and results

For the experimental set-up outlined in Fig. 1, the true atmospheric state $x^{tr}(t)$ starts from a random initial state and is propagated in time by the MSW model using a sample of model parameters from the test set θ^{tr} which are kept constant in time. The objective is to estimate $x^{tr}(t)$ using DA methods and θ^{tr}

using ML methods. To simulate a realistic scenario in our toy model set-up, first 50 DA cycles with n_{ens} forecast and analysis ensemble members were generated without estimating parameters. For the first 50 DA cycles, the forecast model parameters are simply taken from the uniform distributions specified in the previous section. Then, a modified DA cycle takes place which incorporates the ML based parameter estimation (Figure 1: II.-V.). n_{train} parameters are sampled from the uniform distributions, resulting in the set $\{\theta_i^{fc}\}_{i=1}^{n_{train}}$. Each of these parameters represents one training label and is used to generate n_{train} forecast ensemble members (Figure 1: II.). Each forecast corresponds to one training input. Both ML methods are trained on the n_{train} input/label pairs (Figure 1: IV.) before they are used to estimate θ^{tr}. Because the ML algorithms are trained on the full state vector we would ideally use $x^{tr}(t)$ to infer its parameters. Since we do not have access to this state and the observations are sparse with a size smaller than the input size of the ML models we are using the current analysis ensemble (Figure 1: III.). Each analysis ensemble member is used to generate a set of 100 parameter estimates (Figure 1: V.) resulting in a set of $100 \times n_{ens}$ parameters. From this distribution n_{ens} parameters are sampled at each DA cycle for the next 50 cycles, resulting in a different subset $\{\theta_k^{an}\}_{k=1}^{n_{ens}}$ each time. The same experiment is run without the ML modified DA cycle to asses if the parameter estimation is able to reduce the error between the analysis ensemble and the true atmospheric state. Note, in [6] steps IV. + V. were repeated at every DA cycle. Since this is computationally quite expensive with only a small improvement over time we omitted those steps for this present work.

Figure 2 displays the means and standard deviations (std) of the parameter estimates for both ML methods for a training size of $n_{train} = 10000$ and an analysis ensemble size of $n_{ens} = 400$. While the averaged root-mean-square errors (RMSE) of the BNN is slightly smaller than that of the RF, the former tends to be a bit overconfident in its estimates, producing relatively low standard deviations even in cases where the parameter estimates are not very accurate. In this regard, the RF seems to quantify its uncertainty more adequately. Interestingly, there is a visible estimation bias toward intermediate parameter values, which can be observed for all three parameters and for both methods: low parameter values are systematically overestimated and high values are

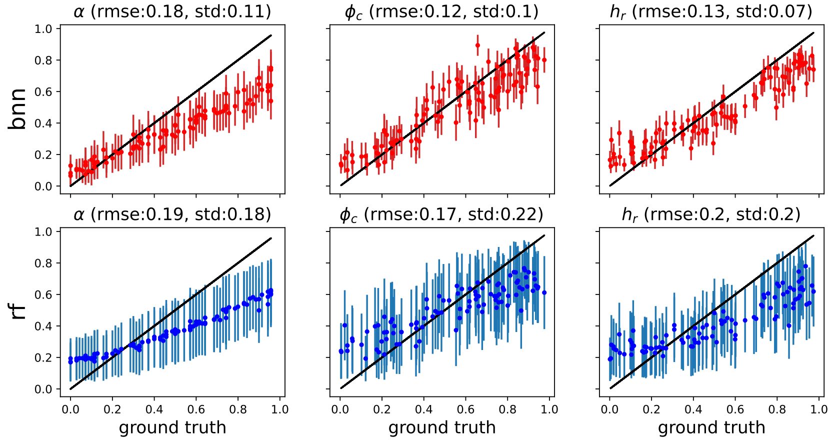

Figure 2: Scatter plot of means and standard deviations of parameter estimates from Bayesian neural network (first row) and random forest (second row) with 400 analysis ensemble members against ground truth for 100 samples of the test set.

underestimated. For the time being, the reason for this bias is not completely clear. We conjecture that it might be caused by the DA procedure we are using, which may not be optimal for convective-scale weather models and produce a discrepancy between forecasts/analysis and the true atmospheric states.

In Fig. 3, the RMSE at the end of the experiment is plotted against the number of ensemble members for the variables of the MSW model and for the parameters. In addition to the methods described above, two other experiments are shown, a first one where true parameter values were used and the state was estimated using DA, and a second one that uses random and false parameter values. For the estimation of parameters, the BNN is more accurate for almost all ensemble sizes. However, standard deviations when using BNN is much lower than RMSE, which is not the case for RF that shows similar values. For atmospheric state, the spread of the ensemble shows that all methods underestimate the RMSE for variables u and h. The situation is different for rain, where estimating parameters increases the uncertainty of the rain field to values higher than RMSE. This is the case for both, the estimation with BNN and RF.

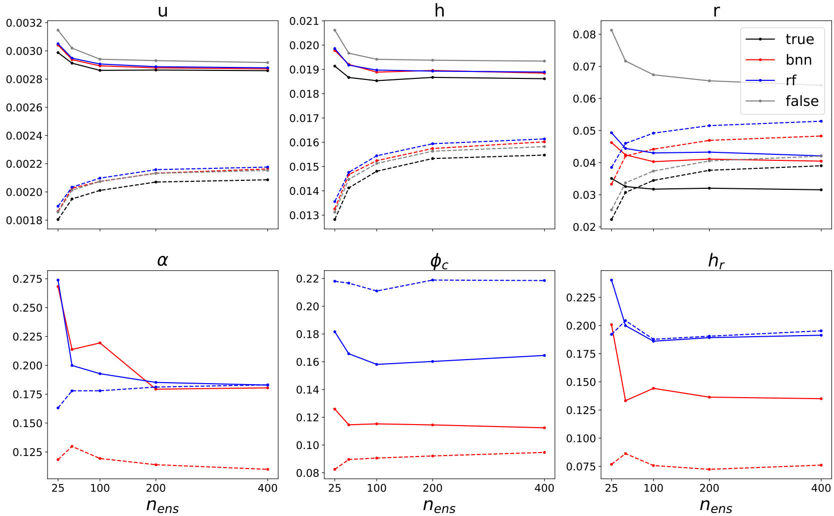

Figure 3: RMSEs (solid) and standard deviations (dashed) of analysis of atmospheric variables (first row) and parameter estimates (second row) against number of analysis ensemble members averaged over last 50 DA cycles and over 100 simulations with different ground truth parameters with $n_{train} = 10000$.

Finally, for the fixed ensemble size of 100, we show the sensitivity of the results to the number of training samples in Fig. 4. As seen there, for 100 members BNN needs at least $n_{train} = 5000$ to outperform RF for two parameters, while not even 10000 samples are enough for the rain removal rate α.

4 Conclusion

In this work, we compared Bayesian neural networks [5] and random forests for the estimation of parameters of the one-dimensional MSW model as a function of the analysis of the atmospheric state. Through perfect model experiments we show that both approaches are in principle able to estimate model parameters and to quantify the related uncertainty. However, while BNN seems to produce more accurate results on the test problems, the uncertainty estimates of RF are closer to RMSE values. For both methods, we observed a systematic estimation bias in boundary regions where parameters are very low or very high.

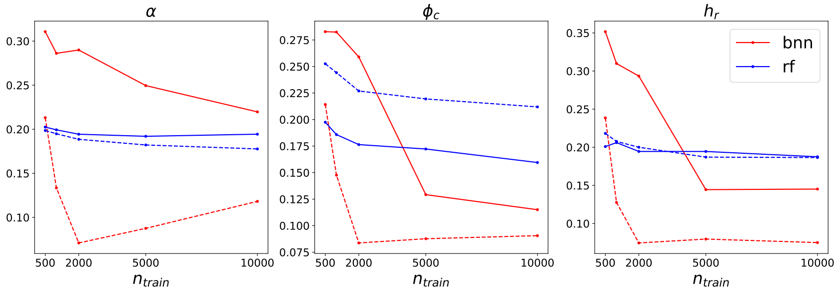

Figure 4: RMSEs (solid) and standard deviations (dashed) of parameter estimates against number of training samples averaged over 100 simulations with different ground truth parameters with $n_{ens} = 100$.

Moreover, the estimation of parameters combined with DA for the state decreases the initial state errors even when assimilating sparse and noisy observations.

References

[1] E. Bingham, J.P. Chen, M. Jankowiak, F. Obermeyer, N. Pradhan, T. Karaletsos, R. Singh, P.A. Szerlip, P. Horsfall, N.D. Goodman. "Pyro: Deep Universal Probabilistic Programming". In: *J. Mach. Learn. Res.,* 20:1-6, 2019.

[2] L. Breiman. "Random Forests". Machine Learning, 45(1):5–32, 2001.

[3] G. Evensen. "The Ensemble Kalman Filter: Theoretical Formulation and Practical Implementation". In: *Dynamics,* 53:343-367. 2003.

[4] M.D. Hoffman, D.M. Blei, C. Wang, J. Paisley. "Stochastic Variational Inference". In: *J. Mach. Learn. Res.,* 14:1303–1347. 2013.

[5] L.V. Jospin, H. Laga, F. Boussaid, W. Buntine, M. Bennamoun. "Hands-On Bayesian Neural Networks—A Tutorial for Deep Learning Users". In: *IEEE Computational Intelligence Magazine,* 17:29-48. 2022.

[6] S. Legler and T. Janjic. “Combining Data Assimilation and Machine Learning to Estimate Parameters of a Convective-Scale Model”. In: *Quarterly Journal of the Royal Meteorological Society,* 148(743):860–874. 2021.

[7] Y.M. Ruckstuhl and T. Janjic. “Parameter and State Estimation with Ensemble Kalman Filter-based Algorithms for Convective-scale Applications”. In: *Quarterly Journal of the Royal Meteorological Society,* 144(826–841). 2018.

[8] M. Würsch and G.C Craig. “A simple Dynamical Model of Cumulus Convection for Data Assimilation Research”. In: *Meteorologische Zeitschrift,* 23(5):483–490. 2014.