Received: 23 July 2021 | Revised: 13 November 2021 | Accepted: 7 December 2021 | Published on: 4 January 2022

DOI: 10.1002/qj.4235

Quarterly Journal of the Royal Meteorological Society RMetS

RESEARCH ARTICLE

Combining data assimilation and machine learning to estimate parameters of a convective-scale model

S. Legler | **T. Janjić**

Meteorological Institute, Ludwig-Maximilians-Universität München, Munich, Germany

Correspondence
S. Legler, Meteorological Institute, Ludwig-Maximilians-Universität München, Theresienstrasse 37, 80333 Munich, Germany
Email: S.Legler@physik.uni-muenchen.de

Funding information
German Research Foundation (DFG) through the 'Waves to Weather' subproject a Heisenberg Award (DFG, Grant/Award Number: JA1077/4-1

Abstract

Errors in the representation of clouds in convection-permitting numerical weather prediction models can be introduced by different sources. These can be the forcing and boundary conditions, the representation of orography, the accuracy of the numerical schemes determining the evolution of humidity and temperature, but large contributions are due to the parametrization of microphysics and the parametrization of processes in the surface and boundary layers. These schemes typically contain several tunable parameters that are either not physical or only crudely known, leading to model errors. Traditionally, the numerical values of these model parameters are chosen by manual model tuning. More objectively, they can be estimated from observations by the augmented state approach during the data assimilation. Alternatively, in this work, we look at the problem of parameter estimation through an artificial intelligence lens by training two types of artificial neural network (ANN) to estimate several parameters of the one-dimensional modified shallow-water model as a function of the observations or analysis of the atmospheric state. Through perfect model experiments we show that Bayesian neural networks (BNNs) and Bayesian approximations of point estimate neural networks (NNs) are able to estimate model parameters and their relevant statistics. The estimation of parameters combined with data assimilation for the state decreases the initial state errors even when assimilating sparse and noisy observations. The sensitivity to the number of ensemble members, observation coverage and neural network size is shown. Additionally, we use the method of layer-wise relevance propagation to gain insight into how the ANNs are learning and discover that they naturally select only a few grid points that are subject to strong winds and rain to make their predictions of chosen parameters.

KEYWORDS

Bayesian neural network, convective-scale data assimilation, EnKF, layer-wise relevance propagation, parameter estimation

1 | INTRODUCTION

In recent years machine learning (ML) has become a subject of interest in various research fields within atmospheric physics. Attempts to including ML in climate and weather modelling range from using it to represent sub-grid processes in global climate models (O'Gorman and Dwyer, 2018; Rasp *et al.*, 2018; Yuval and O'Gorman, 2020), through replacing data assimilation (DA) by an artificial neural network (ANN) to emulate the ensemble Kalman filter (EnKF; Härter and de Campos Velho, 2012), to utilizing an ANN as a surrogate for the complete physical model (Brajard *et al.*, 2020) or for the model error (Farchi *et al.*, 2021) during the DA. Bonavita and Laloyaux (2020) use ANNs to estimate model error tendencies in the Integrated Forecasting System (IFS) of the European Centre for Medium-Range Weather Forecasts (ECMWF) and show that they are able to emulate the main outcomes acquired by the weak-constraint four-dimensional variational (4D-Var) algorithm. Furthermore, computational cost can be improved when including ML in the DA. For example, Ruckstuhl *et al.* (2021) used a convolutional neural network (CNN) to show that a hybrid of a CNN and the EnKF is able to decrease the analysis/background error, equivalent to results obtained by the quadratic programming ensembles (QPEns; Janjić *et al.*, 2014), but with a reduced computational cost compared to that of the QPEns. Finally, ML approaches can also be improved with DA methods by replacing the back-propagation during the training with an adaptive EnKF (Trautner *et al.*, 2020).

While there are many examples of using ML to enhance the analysis/forecast of the model state, deep learning for model parameter estimation is not well developed, especially on the convective scale. In Yadav *et al.* (2020) the coupling parameter of the two-level Lorenz'96 model (Lorenz, 2005) is estimated as a function of the resolved, large-scale state variable using a Gaussian Process (GP; Rasmussen and Williams, 2006). The GP was compared to two types of ANN and a simple linear regression and outperformed the other methods in most of the experiments. Similarly, DA has been successfully used in geosciences for estimation of the state from sparse and noisy observations. However, when parameters are jointly estimated with the state, several problems arise. For example, parameters are not directly observed and therefore updated through cross-correlations which might not be accurate; parameter values often need to be within certain bounds therefore Gaussian assumptions of DA algorithms are not valid, and finally, to use DA for parameter estimation, the stochastic model for the parameters needs to be pre-specified to restrict the spread in parameters (Ruckstuhl and Janjić, 2018; 2020). In this study, we investigate a possibility of using DA for the state estimation while using ML for parameter estimation in order to overcome some of the problems of the augmented state approach for estimating parameters from observations via DA.

Although ML algorithms show promising results in idealized test cases, in their typical usage they come with two major drawbacks. On the one hand, point estimate ANNs typically do not provide an uncertainty with their predictions, which makes it hard to ascribe a confidence when using them in operational settings. On the other hand, they are still seen as black boxes that do not provide any insight into the functions they are trying to approximate. To tackle the latter problem, Toms *et al.* (2020) introduced layer-wise relevance propagation (LRP) to the geosciences, which can be used to visualize how the ANN makes its prediction. Labe and Barnes (2021) utilized this method to disentangle relative influences on regional surface temperatures of aerosols and greenhouse gases in the atmosphere. The former drawback could be approached by using stochastic ANNs instead of their widely used deterministic counterpart. The goal of this study is threefold. First, to estimate parameters of the convective-scale modified shallow-water model from sparse and noisy observations using ML and DA. Second, to compare the predictions and statistics of stochastically trained Bayesian neural networks (BNNs) with an ensemble of deterministically trained point estimate neural networks. And third, to visualize the decision making of the ANNs by applying LRP.

The article is organized as follows. The model and the DA for the state are described in Section 2. Section 3 introduces two ML algorithms for parameter estimation, and their use in combination with DA. This is followed by an investigation of the performance of these two hybrid algorithms in state and parameter space in Section 4. A final discussion and some perspectives are presented in Section 5.

2 | DYNAMICAL MODEL AND STATE ESTIMATION

2.1 | Modified shallow-water model

For this study the same dynamical model, model parameters, and parameter bounds (Table 1) as in Ruckstuhl and Janjić (2018) were used to conduct the experiments. However, instead of using DA with an augmented state approach for parameter estimation, we estimate the parameters with ANNs. In the *twin experiments* presented in this study, the true state (*nature run*) of the atmosphere is generated by the modified shallow-water model (Würsch and Craig, 2014). Synthetic observations are produced by adding random perturbations to the true state. This model is computationally inexpensive but still

TABLE 1 Lower and upper bounds for the uniform distributions of the model parameters

Parameter	Lower bound	Upper bound
α	0.0003	0.001
ϕ_c	899.7	899.9
h_r	90.15	90.25

TABLE 2 Means and standard deviations for the distributions of the observational errors

Variable	Mean	Standard deviation
u	0	0.001
h	0	0.02
r	0.001	1×10^{-7}

represents the key space- and time-scales of storm developments. It is based on the shallow-water equations for the fluid velocity u and the fluid height h with a modification of the geopotential ϕ to include conditional instability. Additionally, a variable for the rain r was added to mimic nature. The equations are:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{\partial(\phi + c^2 r)}{\partial x} = \beta_u + D_u \frac{\partial^2 u}{\partial x^2}, \tag{1}$$

$$\phi = \begin{cases} \phi_c & \text{if } h > h_c, \\ gh & \text{otherwise,} \end{cases} \tag{2}$$

$$\frac{\partial r}{\partial t} + u\frac{\partial r}{\partial x} = D_r \frac{\partial^2 r}{\partial x^2} - \alpha r - \begin{cases} \delta \frac{\partial u}{\partial x} & \text{if } h > h_r \text{ and } \frac{\partial u}{\partial x} < 0, \\ 0 & \text{otherwise,} \end{cases} \tag{3}$$

$$\frac{\partial h}{\partial t} + \frac{\partial(uh)}{\partial x} = D_h \frac{\partial^2 h}{\partial x^2}, \tag{4}$$

where D_u, D_r, D_h are diffusion constants,
$c^2 = g\,h_0$ is the gravity-wave speed for absolute fluid layer $h_0 (h_0 < h_c)$,
δ is the production rate of rain, and α is the removal rate of rain.

Convection is triggered by adding a low-amplitude noise source β_u to the velocity at random locations at every model time step. When the fluid height h exceeds the threshold h_c, which represents the level of free convection, the geopotential is replaced by a lower constant value ϕ_c. The gradient of the geopotential forces fluid to the regions of lower geopotential, which then builds up the fluid height in those regions. Once h reaches the threshold h_r, rain is produced by adding rainwater mass to the geopotential. The removal of rain is mimicked by a linear relaxation towards zero. For the experimental set-up, a one-dimensional grid of length 125 km with 250 grid points was used, which yields a state vector of the form:

$$\mathbf{x} = \begin{bmatrix} \mathbf{u} \\ \mathbf{h} \\ \mathbf{r} \end{bmatrix} \in \mathbb{R}^{750}. \tag{5}$$

The model parameters which were chosen to be estimated are the rain removal rate α, the low constant value for the geopotential ϕ_c and the threshold for the fluid height h_r while the other parameters were known during the experiments. All of the model parameters are global (constant in space), therefore the resulting parameter vector is:

$$\boldsymbol{\theta} = \begin{bmatrix} \alpha \\ \phi_c \\ h_r \end{bmatrix} \in \mathbb{R}^3. \tag{6}$$

Observations are generated from the nature run every 60 model time steps by adding a Gaussian error to u and h and a lognormal error to the r variable to keep it positive. To simulate radar data, the values of u, h, and r are only observed on the grid points where $r > 0.005$. Furthermore, wind observations of 25% of the remaining grid points are added. The model parameters for the nature run are taken from uniform distributions and the size of one model time step is $\Delta t = 4$. The upper and lower bounds of the uniform distributions for the model parameters as well as biases and standard deviations of the observational errors are summarized in Tables 1 and 2 respectively.

2.2 | Stochastic ensemble Kalman filter

Since the focus of this work is testing new algorithms for parameter estimation, a simple stochastic EnKF (Evensen, 1994; 2003) will be utilized for all experiments. It is based on the following cost function for each of the N_{ens} ensemble members:

$$J(\mathbf{x}_t^{a,i}) = (\mathbf{x}_t^{f,i} - \mathbf{x}_t^{a,i})^{\mathrm{T}} \mathbf{P}_t^{-1} (\mathbf{x}_t^{f,i} - \mathbf{x}_t^{a,i}) + (\mathbf{y}_t^i - \mathbf{H}_t \mathbf{x}_t^{a,i})^{\mathrm{T}} \mathbf{R}_t^{-1} (\mathbf{y}_t^i - \mathbf{H}_t \mathbf{x}_t^{a,i}), \tag{7}$$

where $i = 1 \ldots N_{\mathrm{ens}}$ denotes one ensemble member, $\mathbf{x}_t^{f/a,i}$ are the background and analysis states respectively, $\mathbf{R}_t$ is the observation-error covariance matrix and $\mathbf{H}_t$ denotes the observation operator which maps the model states to the observation space. In this work we assume $\mathbf{H}_t$ to be linear. $\{\mathbf{y}_t^i\}$ represents an ensemble of observations acquired by

perturbing the observation vector $\mathbf{y}_t$ such that $\mathbf{y}_t^i = \mathbf{y}_t - \boldsymbol{\epsilon}^i$. $\boldsymbol{\epsilon}^i$ is a perturbation taken from a distribution with a bias and a standard deviation that represent the observation error. The subscript t refers to the time when computation of analysis is being carried out, which usually corresponds to the time appropriate observations are available. For the rest of this section, the subscript t will be omitted. The forecast-error covariance matrix is generated with the ensemble of background states:

$$\mathbf{P} = \overline{(\mathbf{x}^{\mathrm{f},i} - \overline{\mathbf{x}^{\mathrm{f}}})(\mathbf{x}^{\mathrm{f},i} - \overline{\mathbf{x}^{\mathrm{f}}})^{\mathrm{T}}}, \tag{8}$$

where the overline denotes the average over the ensemble members. Minimizing J for each ensemble member yields the analysis ensemble:

$$\mathbf{x}^{\mathrm{a},i} = \mathbf{x}^{\mathrm{f},i} + \mathbf{P}\mathbf{H}^{\mathrm{T}}(\mathbf{H}\mathbf{P}\mathbf{H}^{\mathrm{T}} + \mathbf{R})^{-1}(\mathbf{y}^i - \mathbf{H}\mathbf{x}^{\mathrm{f},i}), \tag{9}$$

with the Kalman gain $\mathbf{K} = \mathbf{P}\mathbf{H}^{\mathrm{T}}(\mathbf{H}\mathbf{P}\mathbf{H}^{\mathrm{T}} + \mathbf{R})^{-1}$. In all experiments exhibited in this study, Equation (9) was used to estimate only the atmospheric state. Once the analysis ensemble and corresponding parameters (Section 3) are estimated, the nonlinear model (1)–(4) would be used to obtain the forecast ensemble $\mathbf{x}^{\mathrm{f},i}$ needed for the next analysis computation.

3 | ML FOR PARAMETER ESTIMATION

The scientific objective of this study is to estimate three model parameters of the modified shallow-water model as a function of the atmospheric state consisting of the atmospheric variables u, h, r using different types of ANN.

3.1 | Types and architecture of ANNs

Two types of ANN are utilized – a *deep ensemble of point estimate neural networks* (NN) and a *Bayesian neural network* (BNN). Both have an input size of 750 – three atmospheric variables for each of the 250 grid points – and an output size of three which corresponds to the three global, unknown parameters that are required to be estimated. The term point estimate neural network is used in this work to refer to the standard type of neural network which, given an input, predicts one deterministic output. To quantify the uncertainty of the estimation produced by the NN, the method of deep ensembles from Lakshminarayanan *et al.* (2017) is adopted. This is an easy implementable approach, where an ensemble of neural networks $\{NN_k\}_{k=1}^{n_{NN}}$ consisting of n_{NN} members with the same architecture but random initial weights is trained independently. The definition of BNNs is not completely consistent across the literature. We adopt its definition from Jospin *et al.* (2020) as a type of ANN "...built by introducing stochastic components into the network..." and trained using *Bayesian inference* (MacKay, 1992). Stochastic components can either be introduced as probability distributions over the activation functions or over the weights, although for this study the latter approach is utilized as this is the more common one. For a more detailed account about the differences between NNs and BNNs and how to utilize BNNs, we refer to Jospin *et al.* (2020). For both ANNs (Tables 3 and 4), fully connected layers were chosen with additional batch normalization layers to accelerate the training (Ioffe and Szegedy, 2015). For the NN, a dropout layer was added to reduce overfitting (Labach *et al.*, 2019) which was not necessary for the BNN. The *Rectified Linear Unit* (ReLU) was used as the activation function for all hidden layers of the NN. This was not possible for the BNN as it resulted in the *dying ReLU problem* (Lu *et al.*, 2020) which is a widely known phenomenon where ReLU neurons output 0 for all inputs. To combat this, the *LeakyReLU* was utilized for all hidden layers of the BNN. The activation functions are defined as:

$$ReLU(x) = \max(0, x), \tag{10}$$

$$LeakyReLU(x) = \max(0, x) + 0.01 \min(0, x). \tag{11}$$

The numbers of neurons for the hidden layers were optimized independently for the NN and the BNN and therefore differ from each other.

TABLE 3 Architecture and training specifics of the *NN*

Architecture		
Type of layer	**Size (input × output)**	**Activation function**
Fully connected	750 × 31	ReLU
Batch-norm	31 × 31	None
Dropout ($p = .5$)	31 × 31	None
Fully connected	31 × 19	ReLU
Fully connected	19 × 11	ReLU
Fully connected	11 × 3	None
Training		
Optimizer	**Adam**	
Mini-batch size	32	
Number of epochs	150	

TABLE 4 Architecture, stochastic model and training specifics of the *BNN*

Architecture				
Type of layer	**Size (input × output)**	**Activation function**	**Stochastic Model**	
Batch-norm	750 × 750	None	Priors $p(W_i^{(k,l)})$	$\mathcal{N}(0,1)$
Fully connected	750 × 20	LeakyReLU	Variational distributions $q_\phi(W_i^{(k,l)})$	$\mathcal{N}(\mu,\sigma)$
Fully connected	20 × 20	LeakyReLU	**Training**	
Fully connected	20 × 20	LeakyReLU	Optimizer	Adam
Fully connected	20 × 3	None	Mini-batch size	32
			Number of epochs	3

3.2 | Data generation and training

To generate the input–output pairs for the training, validation, and test datasets, 100,000 sets of parameters are taken randomly from the uniform distributions specified in Table 1. For each set of parameters Equations (1) to (4) are solved for 1,000 model time steps with a time step discretization of $\Delta t = 4$. Snapshots in time at $t = 1{,}000$ of the state vectors are used as the input of the ANNs. The parameters used to generate those states are the corresponding outputs after rescaling them to [0,1]. From these 100,000 input–output pairs, 90% are used for training, 5% for validation and hyperparameter tuning during the training, and 5% for testing. Additionally, the input samples are augmented during the training by adding perturbations taken from distributions with means and standard deviations corresponding to the observational error specified in Table 2. For each input sample, three perturbed samples are added during the training, resulting in a training size of 360,000. Augmenting the training data by adding the observation error resulted in a significant improvement of the validation loss during the training and in a more realistic setting it might also be necessary to augment the training data by adding a model error. The NN is trained via stochastic gradient descent. Since computing the exact Bayesian posterior of the BNN is usually intractable, we utilize the optimization method of stochastic variational inference (Hoffman *et al.*, 2013) where a set of variational distributions is approximated to the exact Bayesian posteriors during the training. A widely used default for the prior weights of BNNs are normal distributions with mean 0 and standard deviation σ (Jospin *et al.*, 2020). After evaluating the trained weights of the point estimate neural networks, it seemed appropriate to set $\sigma = 1$. To simplify training, the variational distributions are initialized as normal distributions as well such that during the training only the means and standard deviations have to be optimized. For both ANNs an adaptable learning rate (Adaptive Moment Estimation, Adam; Kingma and Ba, 2015) was chosen with an initial value of 0.001. The NN is implemented and trained using only the library PyTorch (Paszke *et al.*, 2019), while for the BNN the probabilistic programming language Pyro (Bingham *et al.*, 2019), which is built on PyTorch, is used.

3.3 | Combining DA and ML

All DA experiments presented in this study are conducted as twin experiments. The atmospheric state and model parameters of the nature run start from a sample taken from the test data-set. The state is propagated forward in time using the modified shallow water model while the parameters are kept constant. The background ensemble members start from different states. Whenever a DA cycle is performed according to Equation (9) the model parameters are estimated according to one of the following set-ups.

1. **True**. The true values of the parameters are known and used for the background state throughout all DA cycles.
2. **Random**. The true values of the parameters are not known and picked randomly from a the uniform distributions specified in Table 1.
3. **NN**. The parameters are estimated using the observations (DA cycle = 0) or the analysis (DA cycle > 0) as input to an ensemble of NNs with 15 members trained according to Section 3.2.
4. **BNN$_0$**. The parameters are estimated using the observations (DA cycle = 0) or the analysis (DA cycle > 0) as input to a BNN (BNN$_0$) trained according to Section 3.2.
5. **BNN$_0$+BNN$_t$**. The parameters are estimated using the observations as input to BNN$_0$ (DA cycle = 0) or to BNN$_t$ (DA cycle > 0) trained online according to Section 3.3.1.

The parameter estimates are then used as parameters for the forward model simulations with the modified shallow-water model for the next 60 model time steps until the next DA cycle is performed, and parameters are estimated again with the ML algorithm.

3.3.1 | Remarks

DA cycle = 0: For the first DA cycle, the observations taken from the nature run are used as inputs for the ANNs. If a variable is only partially observed, the unobserved grid points are interpolated with a quadratic interpolation for u and h. For r the unobserved grid points are simply set to 0.

NN: At DA cycle > 0, each analysis ensemble member $\mathbf{x}_t^{\mathrm{a},i}$ is used as input for each NN ensemble member resulting in $15\,N_{\mathrm{ens}}$ parameter estimates. To obtain N_{ens} parameter vectors out of this ensemble, a beta distribution is fitted to the ensemble of parameter estimates. The parameters for each state ensemble member is then taken from this beta distribution. Instead of fitting a beta distribution, one could simply use the mean of the 15 NN parameter estimates for each state analysis member. We conducted the time evolution experiments in Section 4.3 for both methods (only beta is shown) for three different NN ensemble sizes and found that the mean method and small NN ensemble sizes resulted in poor RMSEs for the rain while the RMSEs of u and h did not show a strong sensitivity to the choice of distribution or the NN ensemble size.

BNN$_t$: This set-up is chosen to test the feasibility of online training during the DA cycle with a realistic number of forecast/analysis ensemble members. Since the training size of BNN$_t$ is much smaller than that used for BNN_0, it is necessary to reduce the number of trainable weights for BNN_t. The first modification is to reduce the input size. Experiments from Ruckstuhl and Janjić (2018) indicate that the rain r and fluid height h are stronger correlated to the parameters than the wind u. Hence, instead of using all three atmospheric variables as the input, only r and h are used. Since BNN$_t$ is trained from scratch at each DA cycle, the input size can be left variable. This allows us to train only on those grid points that are actually observed. Additionally, because we assume the true parameters to be constant over the whole grid, it might not be necessary to use all observed grid points as input. Therefore if more than 62 gridpoints (about 25% of the whole grid) are observed, only those 62 grid points with the highest observed values for r are used. We selected these features for the online training after conducting various experiments with different sets of input features while examining their performance. In Equation (12), $()^*$ refers to this reduced state. To further reduce the number of learnable weights, the number of neurons per hidden layer is decreased from 20 to 2. In total, this results in a maximum of around 540 learnable weights. The resulting input for the training is then given by

$$\left(\mathbf{H}_t\mathbf{x}_\tau^{\mathrm{f},i} + \boldsymbol{\epsilon}^i\right)^* \tag{12}$$

with the observation operator $\mathbf{H}_t$ and $i = 1, \ldots, N_{\mathrm{ens}}$. The ten previous points in time $\tau = t-9, \ldots, t$ are used to increase the training size from N_{ens} to $10N_{\mathrm{ens}}$. The labels for these inputs are simply the model parameters $\boldsymbol{\theta}_{t-60}^i$ from the previous estimation where i refers to the ith ensemble member and $t-60$ to the time of the last DA cycle. Noise corresponding to the observational error specified in Table 2 was added during the training for the same reasons as for the training from Section 3.2. The stochastic model, optimizer, and mini-batch size are the same as for BNN$_0$, but nine training epochs were necessary to reach a minimum in the validation loss, which is most likely caused by the reduced training size. The same online training was attempted with the NN which resulted in poor parameter estimates. This was probably caused by the point estimate NNs needed for large training sets.

4 | RESULTS

4.1 | Diagnostics

Besides the standard diagnostic tools – Root Mean Squared Error (RMSE), ensemble spread (spread), and Coefficient of Determination (R^2) – we utilize *layer-wise relevance propagation* (LRP) in this study. LRP is a visualization tool, which takes a trained ANN and an ANN input sample as the input and produces a *LRP heatmap* as the output. The LRP heatmap is a vector of the same size as the ANN input and those entries with higher numerical values can be interpreted as being more relevant for the ANN's prediction than the ones with lower values. This method has been introduced first to the field of computer vision by Bach *et al.* (2015) and the *PyTorch* implementation used in this study is from Böhle *et al.* (2019). For an in-depth explanation of the algorithm, we refer to Toms *et al.* (2020), who recently introduced LRP to the geosciences. The underlying idea of LRP is to calculate a *relevance* for each input pixel by taking a specific ANN output, which in this case would be either α, ϕ_{c} or h_r, and propagating it back through the network according to a certain set of propagation rules. Applying LRP to the ANNs trained in this study could thus give insight into which grid points and atmospheric variables are most relevant for each of the three parameters.

4.2 | Performance

For the first performance evaluation, 500 samples from the test dataset were used to estimate the parameters. For these experiments we assume a fully observed grid and no

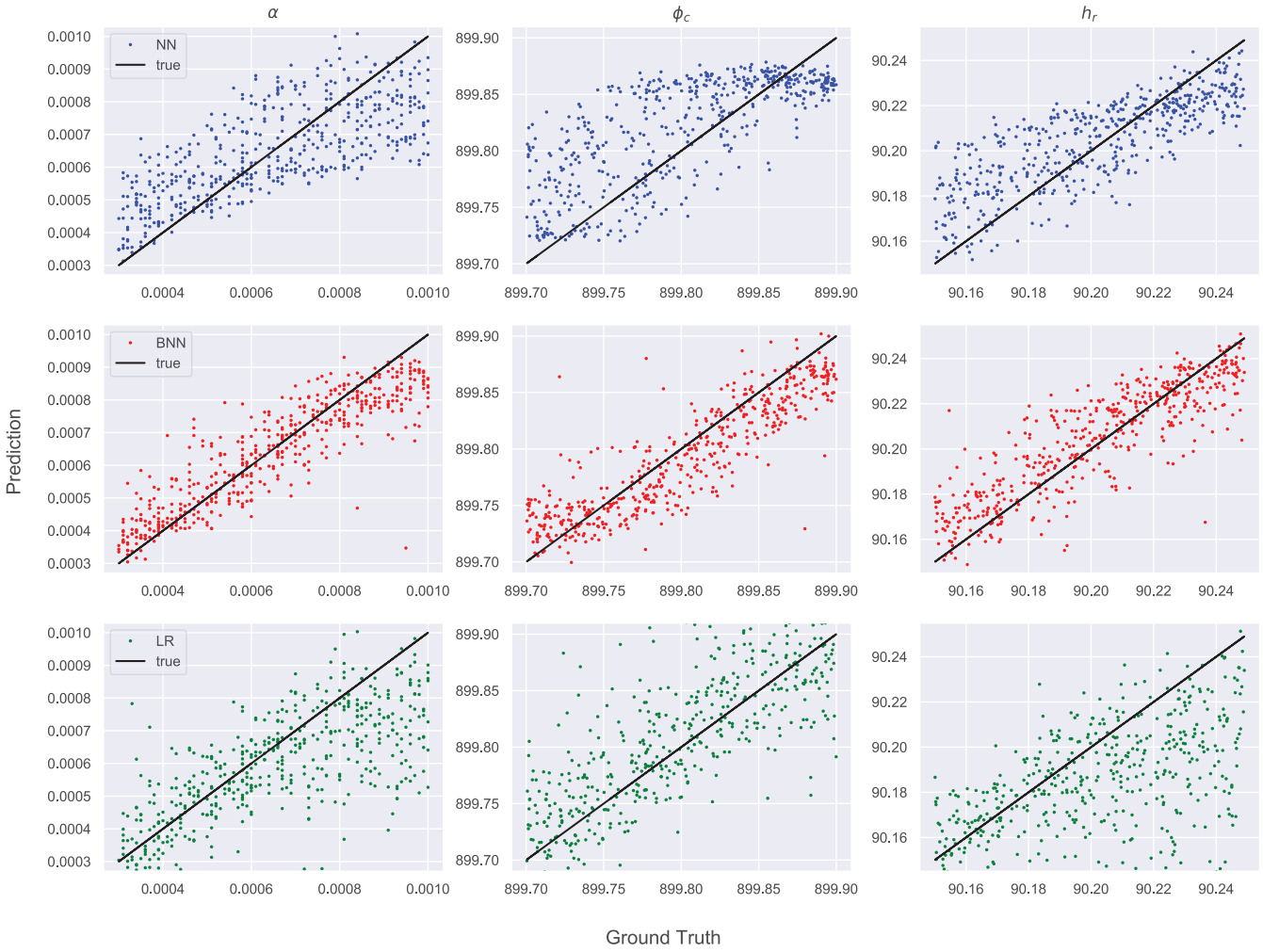

FIGURE 1 Output of NN (blue dots), BNN (red dots), and LR (green dots) against corresponding ground truths and ideal output (black lines) of 500 samples

observation noise. Each output ensemble was averaged and then plotted against its corresponding ground truth (Figure 1). For these experiments no DA was used. Additionally, as a baseline model a simple linear regression (LR) model was fitted to the same training data. As a benchmark, the ideal output was plotted as well, which corresponds to the black lines with slope 1. The BNN outperformed the NN as well as the LR in all three parameters while the LR had the lowest R^2 scores for all parameters (Tables 5 and 6). While the BNN had similar R^2 scores for the different parameters, the performances of NN and LR varied greatly between them. For h_r the LR performed even worse than a baseline model which would predict the average value of the parameter bounds for all inputs. The scatter plot in Figure 1 emphasizes that both ANNs slightly overestimate low parameter values while underestimating high ones, while the LR predicts values that are substantially out of bounds for all parameters.

TABLE 5 R^2 of the parameter predictions plotted in Figure 1 for NN, BNN and LR

Model	α	ϕ_c	h_r
NN	0.53	0.44	0.62
BNN	0.79	0.74	0.75
LR	0.41	0.26	−0.52

4.3 | Time evolution

The remaining experiments presented in this section are conducted according to Section 3.3 and, if not stated otherwise, averaged over 100 individual experiments with different ground truth values. Since NN and BNN_0 are trained on snapshots of model states from one point in time, the question arises how they perform when using states from later points in time as the input and comparing them with BNN_t, which is constantly retrained

TABLE 6 Averaged error relative to the width of the bounds from Table 1, in % of the parameter predictions plotted in Figure 1 for NN, BNN and LR

Model	α	ϕ_c	h_r
NN	16	18	14
BNN	9	11	10
LR	17	19	26

over time. If the predictive power of the former does not significantly decrease, it would only be necessary to train the ANNs once and they could then be used to predict parameters whenever necessary; this would be computationally cheap. In Figure 2 the parameter RMSEs and parameter ensemble spreads of all ANNs are plotted against time in DA cycles. The first DA cycle (DA cycle = 0) starts after running the nature run as well as all background state ensemble members for 1,000 model time steps. The RMSE of the parameters is smallest at the beginning of the time evolution. After that, they grow for about 50 cycles and then oscillate around a relatively constant value (Figure 2). Although BNN_0 outperforms the NN for the initial estimate at DA cycle = 0, over time the RMSEs increase a lot faster for BNN_0 than for NN. For BNN_0+BNN_t, the RMSEs also increase over time, but to a lesser extent than the other methods. While the parameter spreads of BNN_0 and BNN_0+BNN_t increase, which is in accordance with the increased RMSEs, the spreads of α and h_r of NN decrease within the first few DA cycles (Figure 2). Additional experiments are conducted with the online-trained BNN where the initial parameter estimates of BNN_0 are perturbed (Figure 2, orange) to check if BNN_t still performs better than the offline method when starting from a worse initial estimate. We find that the initial RMSEs of the parameters decrease if enough state ensemble members are used (here 400). The sensitivity of BNN_t to the state ensemble size is further discussed in Section 4.5.

4.4 | Distribution

Histograms of the parameter estimation of h_r for one single experiment were plotted for the NN as well as for both BNN methods (Figure 3). Since this is only a single experiment, the results shown here are not statistically significant. However, they still illustrate the key differences between the methods. To also investigate the change of the distributions over time, a histogram for each method was

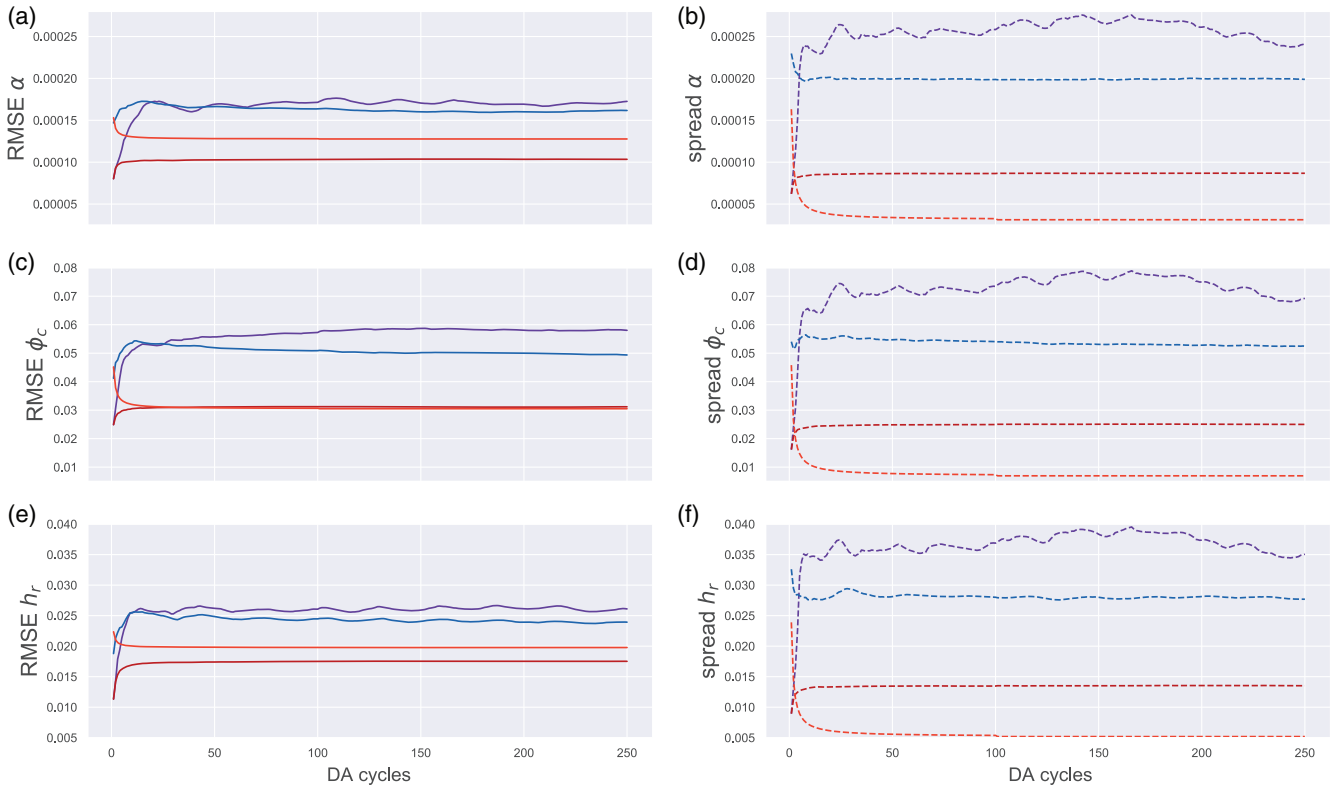

FIGURE 2 Time evolution of (a, c, e) RMSE and (b, d, f) ensemble spread of parameter estimates (a, b) α, ϕ_c and (e, f) h_r against time in DA cycles with 25 analysis ensemble members for NN (blue), BNN_0 (purple), BNN_0+BNN_t (red), and with 400 ensemble members for $BNN_{0,pert}$+BNN_t (orange) averaged over 100 experiments

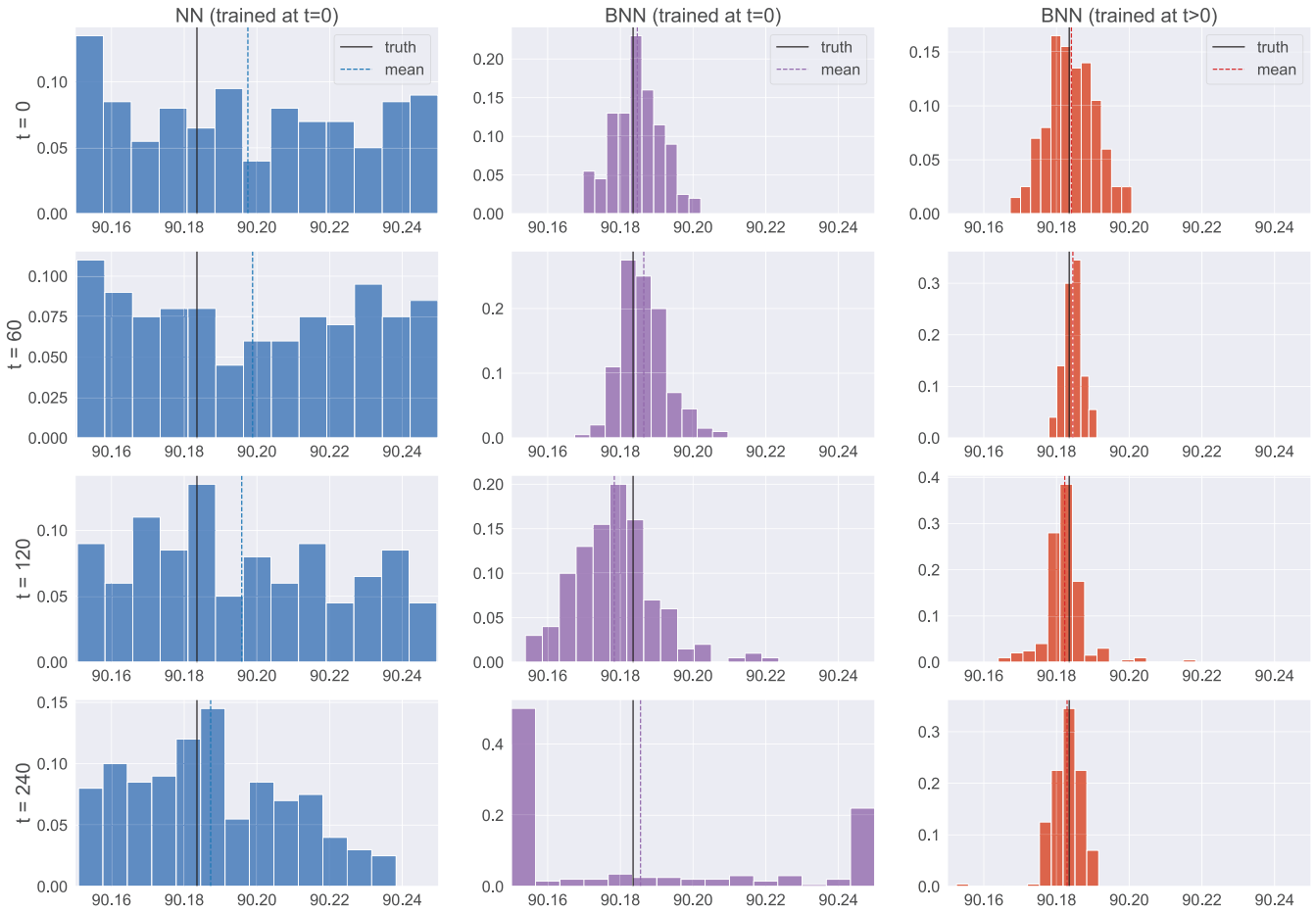

FIGURE 3 Probability histograms of estimates of rain threshold h_r from one single experiment for NN (left), BNN_0 (middle), and BNN_0+BNN_t (right) over time with 200 ensemble members

plotted from t = 0 (DA cycle = 0) to t = 240 (DA cycle = 4). While the NN estimates are spread out over the whole range, the estimates of the BNNs are close to Gaussian distributions with the mean near the true value and a small variance. Nonetheless, over time the prediction of BNN_0 spreads out and it loses its predictive power for DA cycle ≥ 4. Since BNN_0 starts predicting parameters that lie greatly outside the bounds, it was necessary to map the outliers to the bounds as otherwise the modified shallow-water model could not run, which explains the aggregations near the edges. Since this was not necessary for the other two methods, the remaining experiments were only conducted using NN and BNN_0+BNN_t.

4.5 | Sensitivities

The sensitivity of the RMSE and ensemble spread of the atmospheric variable and the model parameter estimates to the number of state ensemble members (Figure 4) and observation coverage (Figure 5) is studied for NN and BNN_0+BNN_t to compare their performance and statistics with the best (black) and worst (grey) case scenarios and to investigate the capabilities of the ANNs under sparse and noisy conditions. For all methods, 100 experiments with 250 DA cycles each are conducted and averaged over the last 100 DA cycles. As expected, the RMSEs of the atmospheric variables decrease for all set-ups with an increase in state ensemble members (Figure 4). The BNN has lower RMSEs than the NN in all experiments and achieves results close to the best-case scenario for u and h for a large ensemble size of 400. However, the NN is more beneficial to the RMSE/spread ratio than the BNN which is likely due to the small parameter spread of the BNN.

The same experimental set-up using the augmented state approach with the stochastic EnKF was studied and displayed in Figure 8 (left column) of Ruckstuhl and Janjić (2018). Although the numerical values of the state errors are not directly comparable due to different implementations of the modified shallow-water model, we can compare how close each method comes to the respective ideal case ("true"). For u and h as well as for small ensemble sizes in r, the augmented state approach exhibits RMSEs which are very close to or even smaller than the case

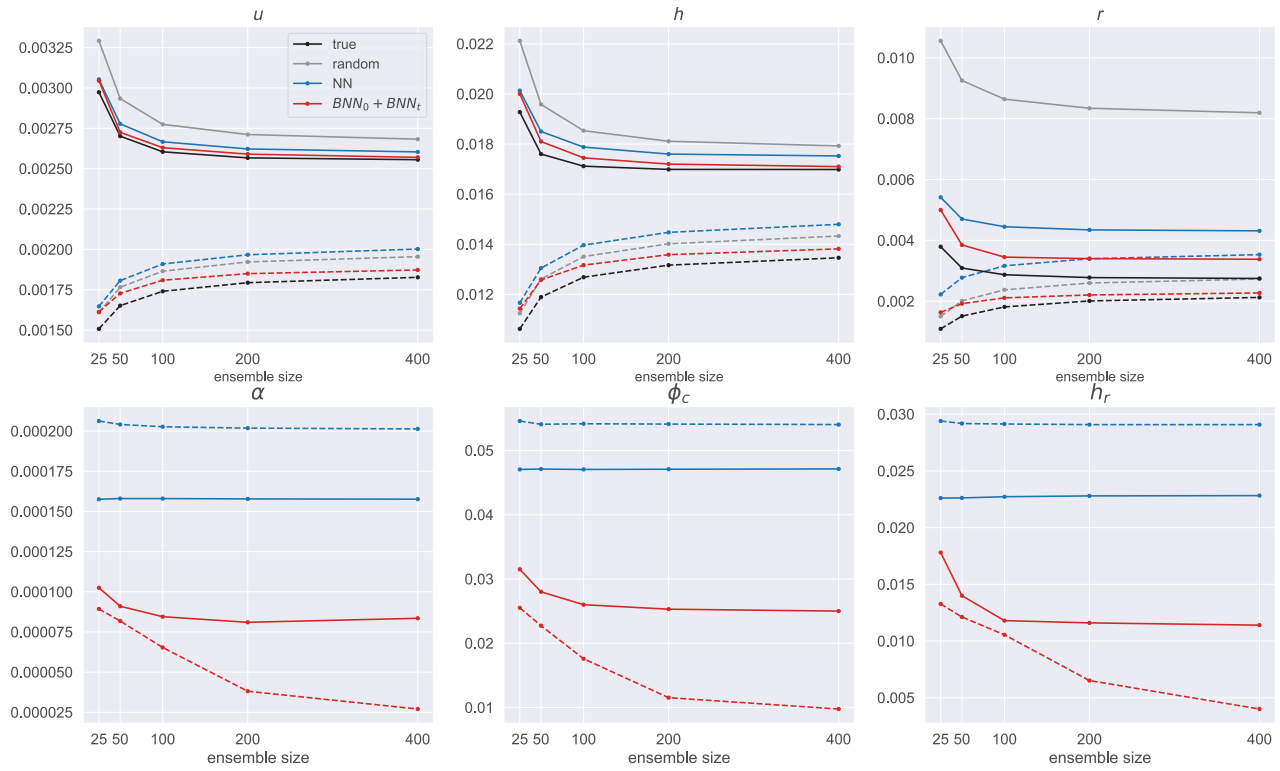

FIGURE 4 RMSE (solid) and ensemble spread (dashed) of atmospheric variable estimates (upper row) and parameter estimates (lower row) against analysis ensemble size averaged over last 100 DA cycles of 100 experiments

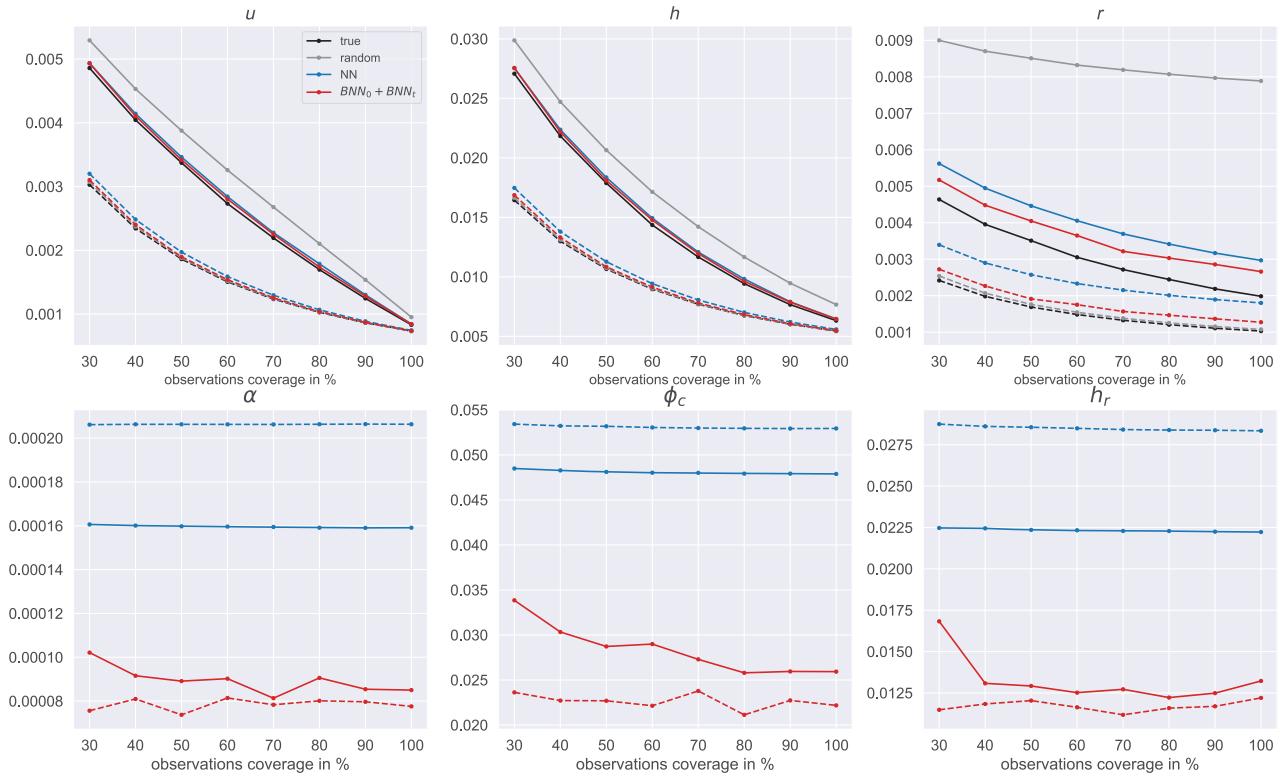

FIGURE 5 RMSE (solid) and ensemble spread (dashed) of atmospheric variable estimates (upper row) and parameter estimates (lower row) against observation coverage with 50 ensemble members averaged over last 100 DA cycles averaged over 100 experiments

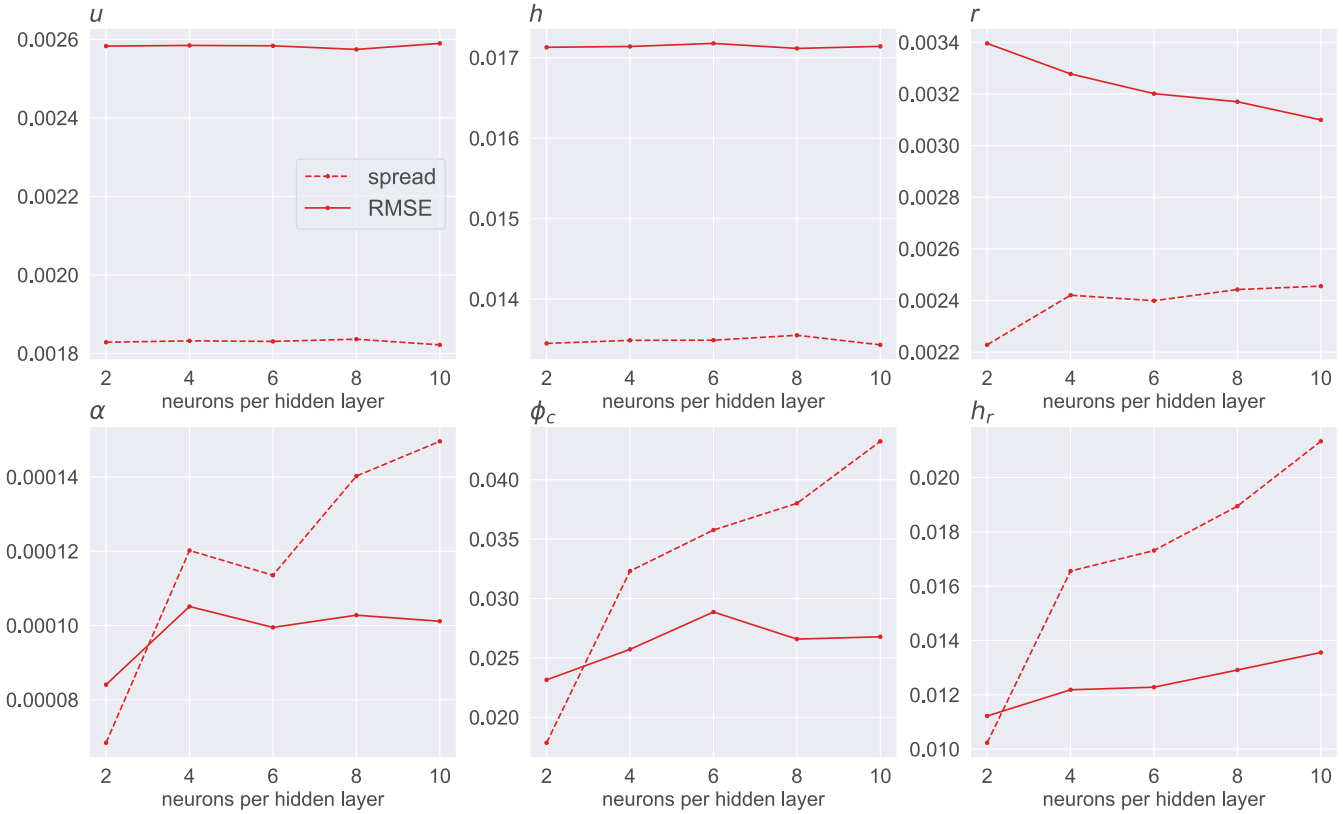

FIGURE 6 RMSE (solid) and ensemble spread (dashed) of atmospheric variable estimates (upper row) and parameter estimates (lower rew) against network size in neurons per hidden layer averaged over last 100 DA cycles of 100 experiments using BNN_0+BNN_t with 200 state ensemble members

without parameter error and outperforms both ANNs. For larger ensemble sizes, the error reduction of BNN_0+BNN_t is greater than that of the augmented state approach for r and about the same for u and h. It should be noted that, even though NN performs worse than BNN_0+BNN_t as well as the augmented state approach in all experiments, its computational cost during the DA cycle is cheaper than that of the other two approaches.

The parameter estimates of the NN do not exhibit any sensitivity to the state ensemble size, while the BNN's RMSEs as well as its spreads decrease with more ensemble members. The sensitivity of the BNN can be explained due to the ensemble size directly controlling the training size and more training data usually increases the predictive capabilities of ANNs and leads to smaller epistemic uncertainties (Der Kiureghian and Ditlevsen, 2009). The very small spreads for large ensemble sizes could be a sign that the BNN is underfitting (Depeweg *et al.*, 2018) in these experiments due to the very small network size of only two neurons per hidden layer and might positively be influenced by increasing the neurons of the hidden layers. To test this hypothesis, network size sensitivity experiments are conducted in Figure 6. For these results the same 100 experiments as before are conducted with the state ensemble size set to 200 and varied neurons per hidden layer. The parameter RMSEs increase slightly with an increase in neurons, as does the spread. The large spreads of the parameter estimates for large network sizes indicate that in these cases the BNN is overfitting (Depeweg *et al.*, 2018). We hypothesize that for each analysis ensemble size there is an optimal BNN network size such that the over- and underfitting effects are minimized. However, increasing the network size of the BNN surprisingly has a positive effect on the rain r where the RMSE decreases while its spread increases. The velocity u and fluid height h on the other hand show no sensitivity to the network size. Figure 6 indicates that not only the accuracy of the parameter estimation but also its spread is relevant for the state error and a slightly overfitted BNN is preferable for the rain. This is probably caused due to the fact that rain production in the modified shallow-water model is largely controlled by the parameters estimated here which act as on–off switches. If the parameter estimates are under-dispersive, there are a lot of grid points where it is raining too early or too late, which in turn results in higher RMSEs of the rain.

The NN and BNN_0 are trained on the full grid, but observations in real-life settings are usually sparse.

Therefore, their sensitivity to the *observation size* was investigated, which is here defined as the percentage of observed grid points. Instead of observing only those grid points whose rain values exceed a certain threshold (as in the previous experiments), percentages corresponding to the values of the x-axis in Figure 5 of random grid points were observed. The RMSEs of the atmospheric variables, especially those of u and h, show a strong sensitivity to the observation size and decrease with more observations available. The parameter RMSEs of the NN, on the other hand, show almost no sensitivity while the parameter estimate RMSEs of the BNN generally decrease with higher observation coverage. The sensitivity of the BNN could be caused by the fact that we only train on the observed grid points of h and r which exhibit the highest rain values. If the most relevant grid points of the BNN are rainy grid points, the chance of observing those is higher with more observation coverage.

TABLE 7 LRP relevance (%) of each atmospheric variable for each parameter with NN trained to estimate all parameters mutually

Parameter	u	h	r
α	20	57	23
ϕ_c	23	61	16
h_r	21	63	16

TABLE 8 LRP relevance (%) of each atmospheric variable for each parameter with three individually trained NNs

Parameter	u	h	r
α	16	59	25
ϕ_c	19	44	37
h_r	21	38	41

4.6 | LRP

To investigate which atmospheric variables were most relevant for each parameter estimation, LRP was applied to all three parameters for 500 inputs and averaged. The LRP algorithm described is utilized with the NN only since, to the best of our knowledge, LRP has not been applied to BNNs so far. The inputs used in these experiments are observations taken from the true atmospheric state of fully observed grids. The total relevances in Table 7 are simply the LRP heatmap values of each parameter for a certain atmospheric variable summed and divided by the total sum of LRP heatmap values for that parameter. Even though all experiments conducted so far indicate that r is the most sensitive variable to the parameter estimation, for the NN surprsingly h is the most relevant variable. To check if this is simply due to the fact that the parameters are predicted simultaneously, the same experiments are repeated with three individually trained NNs, one for each parameter. For this, three training and test datasets are generated, each time keeping two of the parameters constant while varying the parameter to be estimated. The relevances for the individually trained NNs in Table 8 shift

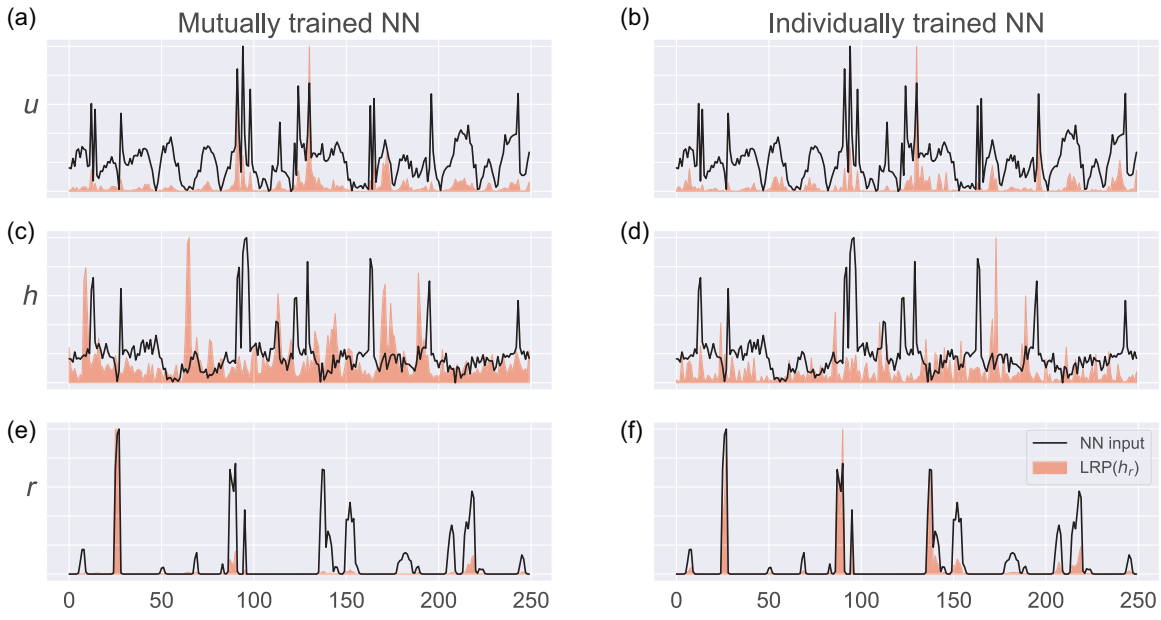

FIGURE 7 Rescaled values (black) of (a, b) fluid velocity u, (c, d) height h and (e, f) rain r, and corresponding LRP values of parameter h_r (red) of (a, c, e) NN trained according to Section 3.2 and of (b, d, f) individually trained NN against all 250 grid points for a single experiment

from the height h towards the rain r for ϕ_c and h_r where h and r are now relatively balanced. For the rain removal rate α, the relevance of h actually increased. To investigate the spatial structure of the LRP relevances and their corresponding inputs, the heatmaps of a single experiment are plotted for the parameter h_r in Figure 7 with both NN set-ups (for the wind velocity u the absolute values were used for easier comparison). These indicate that the NN which is trained to estimate all three parameters mutually (Figure 7a, c, e) uses only a small number of grid points to make its prediction, contrary to the heatmaps of the individually trained NN (Figure 7b, d, f) whose relevances are more spread out over the whole grid for h and r. The heatmaps of u and r look similar to the input indicating that those grid points with strong winds and rain are especially relevant for the NN. However the heatmap of h looks very different from their input. If one plots the LRP heatmap of h together with the rain (not shown), it seems like the relevant grid points of h are the ones where it is in fact raining. This would explain why simply interpolating the observations and using these to estimate the parameters works well in DA cycle = 0 of Figure 2.

5 | CONCLUSION

In this study two types of ANN are trained to estimate the tunable model parameters of the convective-scale modified shallow-water model (Würsch and Craig, 2014). In the perfect model experiments, the NN as well as the BNN are able to decrease the state errors of the atmospheric variables compared to the case were no parameter estimation is applied. The largest reduction of the state error is ultimately found for the rain r. Furthermore, the ANNs investigated here provide tools to quantify the uncertainty of the parameter estimation which increases the ensemble spread of the state analysis and forecast while decreasing their RMSEs. Interestingly, even though the rain exhibits the largest sensitivity on the parameter estimation, the LRP algorithm shows that the fluid height h was the most relevant variable for the NN's prediction. In summary, the BNN produced more accurate estimates while needing less training time and hyperparameter tuning.

In all experiments conducted in this study, the parameters of the nature run are kept constant in time and space. Future work testing the ANNs' ability to estimate local and temporal parameters is therefore required. Furthermore, the input to the ANNs utilized in this study are snapshots of the grid at one point in time. Alternatively, one could investigate the use of different features such as time series of one or more grid points. By utilizing not only atmospheric variables as input features but also "climatological predicators", such as latitude, longitude, time of the day and month, (Bonavita and Laloyaux, 2020) show that the the predictive abilities of the ANNs are greatly enhanced. Providing the ANN with information on the geographical location, diurnal cycle, and seasonal cycle during the training is an interesting approach which could have potential benefits for the parameter estimation problem as well.

It should be noted that the training data, as well as the observations, are generated by the same simplistic model and it is not clear how well the ANNs' predictive ability translates to more complex models and real observations. Before testing them in more realistic scenarios, it is necessary to scale down their demand for large training sizes, possibly by reducing the number of input features or number of hidden layers as demonstrated here. Indeed, the LRP heatmaps show that, if all parameters are estimated simultaneously, the NN makes use of only a few select grid points and that certain atmospheric variables are more relevant than others. Since the online method investigated in this study is accurate but computationally expensive while the offline method is computationally cheap but less precise, it might be possible to have a hybrid approach of both in a more realistic setting. It might not be necessary to retrain at every DA cycle but instead train specialized ANNs for certain seasons/locations/times of day. However, this is an active area of research.

Application of machine learning methods to high-dimensional problems is a challenging task. The idea proposed here would require a large database of simulations with different parameter values. Naturally some of these experiments are done for tuning of the atmospheric models manually as currently done in practice. Next, one would need to produce a stochastic model for parameters with machine learning. With use of BNNs we do not specify statistical properties *a priori* which could make training easier; at least assuming some statistical properties *a priori* might even be necessary for high-dimensional applications. In our approach, the combination with data assimilation allows further online training of the network and further improvements on a ML model. Additionally, when training with a more realistic physical model, one might need to take into account other sources of the model error by either augmenting the training data, which would lead to a larger dataset without having to generate more simulations, or by incorporating the model-error covariance into the loss function.

To improve the ANNs one could investigate not only fully connected layers, as done in this study, but also convolutional or recurrent layers or alternative kinds of stochastic ANNs. Leinonen *et al.* (2021) successfully train a stochastic generative adversarial network (GAN) to downscale time-evolving images of atmospheric fields from low to high resolution. The conditional GAN trained in Leinonen *et al.* (2021) consists of convolutional and

recurrent layers and is able to predict images larger than those it was trained on and can predict longer time series than the sequences used for training. This reduces the need for large training sizes and offers the possibility for offline training. Furthermore, Gagne *et al.* (2020) successfully trained conditional GANs to estimate stochastic parametrizations of the Lorenz'96 model although further studies are required to test if GANs are also suitable for parameter estimations of convective-scale models.

In the studied test case, with perfect model assumptions and enough training data, the ANNs were able to estimate the unknown model parameters and quantify their uncertainty more accurately than a simple linear regression, even under sparse and noisy conditions. Including the parameter estimates obtained from the ANNs in the DA cycle resulted in reduced state errors and increased ensemble spreads compared to the case without parameter estimation and unknown parameters.

ACKNOWLEDGEMENTS

The research leading to these results has been done within the subproject B6 of the Transregional Collaborative Research Center SFB/TRR 165 "Waves to Weather" funded by the German Research Foundation (DFG). T. Janjić is also grateful to the DFG for funding her Heisenberg Award (DFG JA1077/4-1).

CONFLICT OF INTEREST

The authors declare that they have no conflict of interest.

ORCID

S. Legler https://orcid.org/0000-0002-1261-059X
T. Janjić https://orcid.org/0000-0002-8837-0879

REFERENCES

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R. and Samek, W. (2015) On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLoS One*, 10(7). https://doi.org/10.1371/journal.pone.0130140.

Böhle, M., Eitel, F., Weygandt, M. and Ritter, K. (2019) Layer-wise relevance propagation for explaining deep neural network decisions in MRI-based Alzheimer's disease classification. *Frontiers in Aging Neuroscience*, 11, 1–17. https://doi.org/10.3389/fnagi.2019.00194.

Bingham, E., Chen, J.P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P.A., Horsfall, P. and Goodman, N.D. (2019) Pyro: deep universal probabilistic programming. *Journal of Machine Learning Research*, 20(1), 973–978. https://doi.org/10.5555/3322706.3322734.

Bonavita, M. and Laloyaux, P. (2020) Machine learning for model error inference and correction. *Journal of Advances in Modeling Earth Systems*, 12(12). https://doi.org/10.1029/2020MS002232.

Brajard, J., Carrassi, A., Bocquet, M. and Bertino, L. (2020) Combining data assimilation and machine learning to emulate a dynamical model from sparse and noisy observations: a case study with the Lorenz'96 model. *Journal of Computational Science*, 44. https://doi.org/10.1016/j.jocs.2020.101171.

Depeweg, S., Hernández-Lobato, J.M., Udluft, S. and Runkler, T.A. (2018). Sensitivity analysis for predictive uncertainty, pp. 279–284. In: Proceedings of European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, 25–27 April, Bruges, Belgium.

Der Kiureghian, A. and Ditlevsen, O. (2009) Aleatory or epistemic? Does it matter? *Structural Safety*, 31(2), 105–112. https://doi.org/10.1016/j.strusafe.2008.06.020.

Evensen, G. (1994) Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics. *Journal of Geophysical Research*, 99, 10143–10162.

Evensen, G. (2003) The ensemble Kalman filter: theoretical formulation and practical implementation. *Ocean Dynamics*, 53, 343–367.

Farchi, A., Laloyaux, P., Bonavita, M. and Bocquet, M. (2021) Using machine learning to correct model error in data assimilation and forecast applications. *Quarterly Journal of the Royal Meteorological Society*, 147, 3067–3084.

Gagne, D.J., Christensen, H.M., Subramanian, A.C. and Monahan, A.H. (2020) Machine learning for stochastic parameterization: generative adversarial networks in the Lorenz '96 model. *Journal of Advances in Modeling Earth Systems*, 12(3). https://doi.org/10.1029/2019MS001896.

Hoffman, M.D., Blei, D.M., Wang, C. and Paisley, J. (2013) Stochastic variational inference. *Journal of Machine Learning Research*, 14, 1303–1347.

Härter, F.P. and de Campos Velho, H.F. (2012) Data assimilation procedure by recurrent neural network. *Engineering Applications of Computational Fluid Mechanics*, 6(2), 224–233.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: accelerating deep network training by reducing internal covariate shift, pp. 448–456. In: Proceedings of the 32nd International Conference on Machine Learning, Vol. 37, 7–9 July, Lille, France.

Janjić, T., McLaughlin, D., Cohn, S.E. and Verlaan, M. (2014) Conservation of mass and preservation of positivity with ensemble-type Kalman filter algorithms. *Monthly Weather Review*, 142, 755–773.

Jospin, L.V., Buntine, W., Boussaid, F., Laga, H. and Bennamoun, M. (2020) Hands-on Bayesian neural networks – a tutorial for deep learning users. *ArXiv*. abs/2007.06823.

Kingma, D.P. and Ba, J.L. (2015). Adam: a method for stochastic optimization. In: Third International Conference on Learning Representations, 7–9 May 2015, San Diego, CA.

Labach, A., Salehinejad, H. and Valaee, S. (2019) Survey of dropout methods for deep neural networks. *ArXiv*. abs/1904.13310.

Labe, Z.M. and Barnes, E.A. (2021) Detecting climate signals using explainable AI with single-forcing large ensembles. *Earth and Space Science Open Archive*, 40.

Lakshminarayanan, B., Pritzel, A. and Blundell, C. (2017). Simple and scalable predictive uncertainty estimation using deep ensembles, pp. 6405–6416. In: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, 4–9 December 2017, Long Beach, CA.

Leinonen, J., Nerini, D. and Berne, A. (2021) Stochastic super-resolution for downscaling time-evolving atmospheric fields with a generative adversarial network. *IEEE Transactions on Geoscience and Remote Sensing*, 59, 7211–7223. https://doi.org/10.1109/TGRS.2020.3032790.

Lorenz, E.N. (2005) Designing chaotic models. *Journal of the Atmospheric Sciences*, 62, 1574–1587.

Lu, L., Shin, Y., Su, Y. and Karniadakis, G.E. (2020) Dying ReLU and initialization: theory and numerical examples. *Communications in Computational Physics*, 28(5), 1671–1706. https://doi.org/10.4208/cicp.OA-2020-0165.

MacKay, D.J.C. (1992) A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4, 448–472.

O'Gorman, P.A. and Dwyer, J.G. (2018) Using machine learning to parameterize moist convection: potential for modeling of climate, climate change, and extreme events. *Journal of Advances in Modeling Earth Systems*, 10(10), 2548–2563.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S. (2019) Pytorch: an imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 8024–8035.

Rasmussen, C.E. and Williams, C.K.I. (2006) *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press.

Rasp, S., Pritchard, M.S. and Gentine, P. (2018) Deep learning to represent sub-grid processes in climate models. *Proceedings of the National Academy of Sciences*, 115(39), 9684–9689.

Ruckstuhl, Y. and Janjić, T. (2020) Combined state-parameter estimation with the LETKF for convective-scale weather forecasting. *Monthly Weather Review*, 148(4), 1607–1628.

Ruckstuhl, Y., Janjić, T. and Rasp, S. (2021) Training a convolutional neural network to conserve mass in data assimilation. *Nonlinear Processes in Geophysics*, 28(1), 111–119.

Ruckstuhl, Y.M. and Janjić, T. (2018) Parameter and state estimation with ensemble Kalman filter-based algorithms for convective-scale applications. *Quarterly Journal of the Royal Meteorological Society*, 144, 826–841.

Toms, B.A., Barnes, E.A. and Ebert-Uphoff, I. (2020) Physically interpretable neural networks for the geosciences: applications to earth system variability. *Journal of Advances in Modeling Earth Systems*, 12(9). https://doi.org/10.1029/2019MS002002.

Trautner, M., Margolis, G. and Ravela, S. (2020). Informative ensemble Kalman learning for neural structure. In: Dynamic Data-Driven Applications Systems, Third international conference, 2–4 October, Boston, MA. Berlin: Springer.

Würsch, M. and Craig, G.C. (2014) A simple dynamical model of cumulus convection for data assimilation research. *Meteorologische Zeitschrift*, 23(5), 483–490.

Yadav, N., Ravela, S. and Ganguly, A.R. (2020) Machine learning for robust identification of complex nonlinear dynamical systems: applications to earth systems modeling. *ArXiv*. abs/2008.05590.

Yuval, J. and O'Gorman, P.A. (2020) Stable machine-learning parameterization of subgrid processes for climate modeling at a range of resolutions. *Nature Communications*, 11. https://doi.org/10.1038/s41467-020-17142-3.

How to cite this article: Legler, S. & Janjić, T. (2022) Combining data assimilation and machine learning to estimate parameters of a convective-scale model. *Quarterly Journal of the Royal Meteorological Society*, 148(743), 860–874. Available from: https://doi.org/10.1002/qj.4235