Downscaling shallow water simulations using artificial neural networks and boosted trees

Killian Bakong Epoune, Vincent Guinot, Antoine Rousseau, Gwladys Toulemonde

Downscaling shallow water simulations using artificial neural networks and boosted trees

K. Bakong[1], V. Guinot[1,3], A. Rousseau[1,2], and G. Toulemonde[1,2]

[1]Inria LEMON, Inria, Univ. Montpellier, France
[2]IMAG, Univ. Montpellier, CNRS, Montpellier, France
[3]HydroSciences Montpellier (HSM), Univ. Montpellier, CNRS, IRD, Montpellier, France

October 10, 2022

Abstract

We present the application of two statistical artificial intelligence tools for multi-scale shallow water simulations. Artificial neural networks (ANNs) and boosted trees (BTs) are used to model the relationship between low-resolution (LR) and high-resolution (HR) information derived from simulations provided in the learning phase. The two statistical models are analyzed (and compared) through hyper-parameters such as the number of epochs and the network structure for ANNs, and the learning rate, tree depth and number for BTs. This analysis is performed through 4 numerical experiments the input datasets of which (for the learning, validation and test phases) are varied through the boundary conditions of the flow numerical simulation.

The performance of the ANNs is remarkably consistent, regardless of the choice made for the training/validation/testing set. The performance improves with the number of epochs and the number of neurons. For a given number of neurons, a single-layer structure performs better than multi-layer structures. BTs perform significantly better than ANNs in 2 experiments, with an error 10 to 100 times lower and a computational cost 5 to 10 times larger). However, when the validation datasets differ from the training datasets, the performance of BTs performance is strongly degraded, with a modelling error more than one order of magnitude larger than that of ANNs.

Used in conjunction with upscaled flood models such as porosity models, these techniques appear as a promising operational alternative to direct flood hazard assessment from HR flow simulations.

1 Introduction

Shallow water modelling is considered a reference approach for the simulation of free surface flows over natural and artificial topography. Shallow water models provide a satisfactory trade-off between model complexity and accuracy. However, in a number of applications, such as modelling urban floods or flows over microtopography, the level of detail required by an accurate meshing of the geometry makes data collection, treatment and simulation times prohibitive for an operational point of view [32, 38]. For this

reason, substantial research effort has been devoted to developing multiscale modelling approaches over the past two decades. Multiscale modelling involves two steps: (i) upscaling, whereby the flow problem of interest is solved over a coarse, or Low Resolution (LR) space-time grid, and (ii) downscaling, whereby the results of the LR model are used as a basis to reconstruct High Resolution (HR) variables and/or risk indicators. The interest of the approach lies in its computational efficiency: LR models are several orders of magnitudes as fast as HR models, and the downscaling process is usually computationally affordable. Most of the research on multiscale shallow water modelling to date has focused on the development and validation of upscaled shallow water models [6, 9, 10, 22, 23, 24, 25, 27, 29, 43, 35, 41, 42]. The downscaling process has been little explored in comparison, although interest for the approach has started to emerge from the recent literature [39, 8, 14, 36].

Statistical downscaling approaches are widely used in climate / meteorological modelling to retrieve pointwise information from larger scale information [33, 40]. Hybrid methods using Lagrangian systems have been developed for small scale wind simulation [4]. In an urban flooding context [8] a local univariate approach is compared to a global approach whereby HR information is retrieved simultaneously at multiple locations. To do so, problem dimension is reduced via Principal Component Analysis (PCA). The relationship between the LR and HR fields is inferred via neural networks. Note that non-linear regression algorithms have already been applied to such variables and to meteorological variables in a univariate context [7, 11]. However, neural networks are not the only possible machine learning approach. For instance, [14] report the use of Empiricial Orthogonal Decomposition (EOD) in conjunction with sparse Gaussian models.

The conclusions of [8] raise a number of research questions. A number of these are addressed in the present paper:

(Q1) What is the influence of the method used on downscaling accuracy? In [8], ANN were used. To address this question, boosted trees are used in addition to ANN.

(Q2) Do the various methods perform equally well for all types of flow variables, or are some variables downscaled more accurately than others?

(Q3) In [8], the ratio of the HR to LR grid resolutions (called upscaling ratio) was fixed. However, in view of operational applications, it is important to determine how the accuracy of a downscaling method changes with the upscaling ratio.

Multiscale modelling essentially embeds a number of sources for error. Firstly, upscaling an HR model into a LR model implies that the equations, variables, boundary conditions and/or model parameters are aggregated in some way. Such aggregation necessarily induces a kind of filtering error (see [13]). Secondly, the LR model operating on a coarser grid than the HR model, it is likely subjected to larger truncation and numerical errors in the numerical solution process. Thirdly, reverting from the LR to the HR grid via the downscaling process also induces a downscaling error. It is hoped that an IA downscaling algorithm can allow the biases stemming from the first and second step above to be eliminated via appropriate training. In order to eliminate such biases as much as possible, the present feasibility study is carried out under the assumption of so-called perfect upscaling. In other words, the LR solution is not obtained from an upscaled model running independently from the HR model, but from an exact averaging of the HR solution.

The present paper is organised as follows. Section 2 presents the general framework of upscaling and the shallow water model used. Section 3 describes the AI approaches used and the experiment plan for training, validation and method performance assessment. The results are presented and discussed in Section 4. Section 5 is devoted to concluding remarks and research perspectives.

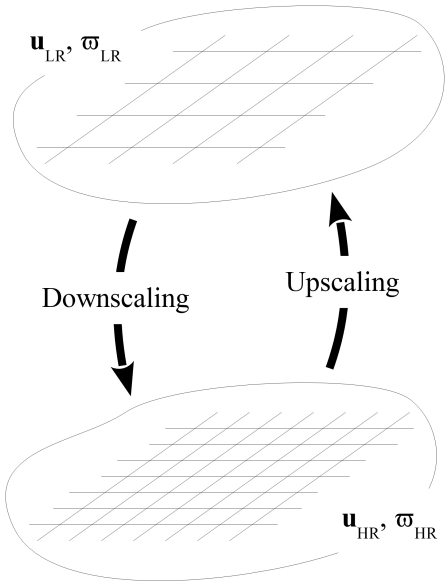

Figure 1: Upscaling and downscaling. Definition sketch.

2 Flow model

2.1 Upscaling and downscaling

Upscaling is usually presented as a filtering problem [13] whereby a Low Resolution (LR) variable vector $\mathbf{u}_{\mathrm{LR}}$ and/or parameter vector $\boldsymbol{\varpi}_{\mathrm{LR}}$ are inferred from High Resolution (HR) variable $\mathbf{u}_{\mathrm{HR}}$ and/or $\boldsymbol{\varpi}_{\mathrm{HR}}$ parameter vectors. Conversely, downscaling is the process of inferring the set $(\mathbf{u}_{\mathrm{HR}}, \boldsymbol{\varpi}_{\mathrm{HR}})$ from the set $(\mathbf{u}_{\mathrm{LR}}, \boldsymbol{\varpi}_{\mathrm{LR}})$ (Figure 1). The HR variable and parameter fields are assumed to satisfy a set of partial differential equations

$$\mathbf{L}_{\mathrm{HR}}\left(\mathbf{u}_{\mathrm{HR}}, \boldsymbol{\varpi}_{\mathrm{HR}}, \mathbf{x}, t\right) = 0 \tag{1}$$

where $\mathbf{L}_*$ $(* = \mathrm{HR}, \mathrm{LR})$ is a differential operator vector, $\mathbf{x}$ and t are respectively the space and time coordinates. Solving Eq. (1) is usually a time-consuming operation. It is therefore preferred to use a significantly faster, LR model

$$\mathbf{L}_{\mathrm{LR}}\left(\mathbf{u}_{\mathrm{LR}}, \boldsymbol{\varpi}_{\mathrm{LR}}, \mathbf{x}, t\right) = 0 \tag{2}$$

and attempt to infer $(\mathbf{u}_{\mathrm{HR}}, \boldsymbol{\varpi}_{\mathrm{HR}})$ from $(\mathbf{u}_{\mathrm{LR}}, \boldsymbol{\varpi}_{\mathrm{LR}})$ via a downscaling operation. The nature of the HR and LR operators, variables and parameters are strongly dependent on the approach and scientific questions addressed. In groundwater flow modelling and transport, it is customary to use the same set of governing equations, flow parameters and flow variables on the HR and LR scales [28, 34]. Upscaling is mostly applied to the hydraulic conductivity and dispesion coefficient, i.e. to the parameter vector. $\boldsymbol{\varpi}_{\mathrm{HR}}$ and $\boldsymbol{\varpi}_{\mathrm{LR}}$ thus have the same structure but take different numerical values. The homogeneization approach, based on multiple scale expansions [1, 2, 3, 12], often uses different structures for $(\mathbf{L}, \mathbf{u}, \boldsymbol{\varpi})$ on the HR and LR scales. In shallow water, porosity-based models [10, 25, 27], the flow variables $\mathbf{u}_{\mathrm{HR}}$ and

$\mathbf{u}_{\mathrm{LR}}$ are the same but the governing equations and the parameter sets are different on the LR and HR scale.
The purpose of the present paper is to test downscaling methods for the shallow water equations, independently of the existence of an upscaled model. To do so, the following approach is retained from [8]: (i) series of HR solutions are obtained by solving Eq. (1) over fine grids, (ii) these HR solutions are filtered using an avering of non-overlapping spatial domains Ω_i to provide the LR solution

$$\mathbf{u}_{\mathrm{LR}}(\mathbf{x},t) = \frac{1}{\Omega_i}\int_{\Omega_i} \mathbf{u}_{\mathrm{HR}}(\mathbf{x},t)\mathrm{d}\Omega_i, \mathbf{x} \in \Omega_i \tag{3}$$

2.2 Shallow water models

2.2.1 Governing equations

The nonlinear shallow water equations were first introduced by Barré de Saint-Venant [REF-1871]. They are a two-dimensional depth-averaged approximation of the incompressible Navier-Stokes equations under the hydrostatic assumption [Ref-Gerbeau-Perthame]. They are widely used in fluid dynamics at various scales, from river hydraulics to ocean-atmosphere dynamics. The 1D shallow water equations are written in conservation form as:

$$\frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{f}(\mathbf{u})}{\partial x} = \mathbf{s}(\mathbf{u}) \tag{4a}$$

$$\mathbf{u} = [h, q]^T = [h, hu]^T \tag{4b}$$

$$\mathbf{f} = [q, M]^T = \left[hu, hu^2 + \frac{g}{2}h^2\right]^T \tag{4c}$$

$$\mathbf{s} = [0, (S_0 - S_f)gh]^T \tag{4d}$$

where g is the gravitational acceleration, h is the water depth, M is the specific force per unit width, q is the unit discharge, S_0 and S_f are respectively the bottom and friction slopes, and u is the flow velocity (see Figure 2).

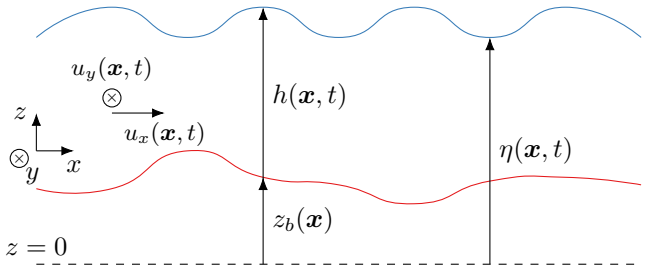

Figure 2: Definition sketch for shallow water models structure and variables. The red and blue lines represent respectively the (steady) bottom elevation and the (unsteady) free surface elevation.

Equations (4a)-(4d) are solved numerically using a finite volume approach. The hyperbolic part of the equations and the source term are computed sequentially using a first-order time splitting procedure [13]. An explicit, Godunov-type [20] shock-capturing method combined with the HLL Riemann solver [26] is used for the solution of the hyperbolic part. In what follows, only frictionless motion over flat bottoms is considered, consequently $\mathbf{s} = 0$.

3 Materials and methods

3.1 Simulation framework

Downscaling is applied to one-dimensional (1D), synthetic shallow water Boundary Value Problems (BVPs). These problems are selected because they are broadly representative of the main features of the shallow water equations (4a)-(4d). The BVPs are solutions of (4a)-(4d) over a domain $(x,t) \in [0,L] \times \mathbb{R}^+$ with the following initial and boundary conditions

$$(h,q)^T(x,0) = (h_0,0) \quad \forall x \in [0,L] \tag{5a}$$

$$h(0,t) = h_1 \quad \forall t > 0 \tag{5b}$$

$$h(L,t) = h_0 \quad \forall t > 0 \tag{5c}$$

The initial conditions correspond to a fluid at rest. When $h_1 \neq h_0$, a rightward travelling wave is initiated at the left-hand boundary ($x = 0$) and propagates into the domain. When the bottom is horizontal and motion is frictionless, the solution is known to be self-similar [30], that is to be a function of the ratio x/t alone. If $h_1 < h_0$, a so-called rarefaction wave appears (Figure 3), with a continuous $(h,q)^T$ profile connecting the tail to the head. The wave propagation direction can be changed easily by swapping the left- and right-hand boundary conditions, replacing Eqs. (5b), (5c) with

$$h(0,t) = h_0 \quad \forall t > 0 \tag{6a}$$

$$h(L,t) = h_1 \quad \forall t > 0 \tag{6b}$$

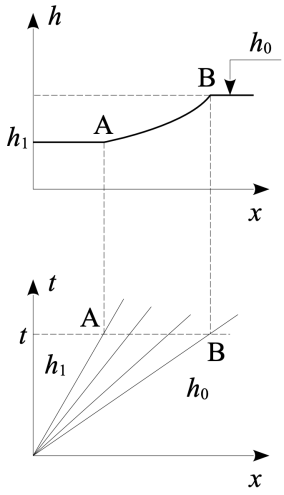

Figure 3: Definition sketch for the experiment plan. Top : representation in the (x,h) plane for a given time t. Bottom : water depth contour lines in the (x,t) plane. Owing to solution self-similarity, the speeds of Points A and B are constant, hence the straight, h-contour lines in the (x,t) plane (bottom).

3.2 Machine Learning (ML) methods

The two ML methods studied in this work are Artificial Neural Networks (ANN) and Boosted Trees (BT). For each method, the non linear algorithm is trained to infer a relationship between the LR and HR profiles. Here, a "profile" is defined as the set of variables of interest within all the cells of the domain at a given time. Let d_{HR} and d_{LR} be respectively the number of cells in the HR and LR grids. $X_k \in \mathbb{R}^{d_{\mathrm{LR}}}$ and $Y_k \in \mathbb{R}^{d_{\mathrm{HR}}}$ denote respectively the LR and HR profiles, that is the vectors containing the variables of interest over the LR and HR grids at the kth time step.
The purpose is to establish a relationship in the form

$$\widehat{Y}_k = \widehat{f}(X_k) \quad \forall k = 1, \ldots, K \tag{7}$$

where $\widehat{Y}_k$ is the reconstructed Y_k and $\widehat{f}$ is a function to be determined. The ML algorithms are trained from the $(X_k, Y_k)_{k=1,\ldots,K}$ to establish $\widehat{f}$ by minimizing the Mean Squared Error (MSE) between the ML-reconstructed data and the HR profile computed by the shallow water model:

$$\mathrm{MSE} = \frac{1}{d_{\mathrm{HR}} K} \sum_{j=1}^{d_{HR}} \sum_{k=1}^{K} \left(\widehat{Y}_k^{(j)} - Y_k^{(j)} \right)^2 \tag{8}$$

where $Y_k^{(j)}$ and $\widehat{Y}_k^{(j)}$ are the jth components of the vectors Y_k and $\widehat{Y}_k$.
The first ML method used in the present work is the ANN approach, one the most widespread deep learning systems (see [21, 31, 37, 5]). Deep learning methods allow computational models composed of multiple processing layers aiming to learn a mapping between inputs and outputs. We consider feed-forward ANN, also known as multilayer perceptrons. The considered scheme contains one input layer, one or more hidden layer(s) with a hyperbolic tangent activation function, and one output layer. The various ANN structures explored in the present work are detailed in Subsection 3.3.2. The stochastic gradient descent method [21] is used to minimise the MSE (8).
The second method is the BT method, a sequential approach based on random trees. BT consist in combining base classifiers, that are trained sequentially on different data sets. The combination weights are defined from the performance of the classifiers (see [15, 16, 17, 18, 19]). As detailed in Subsection 3.3.2 the influence of two parameters is investigated: the maximum depth of each tree and the minimum allowed samples per leaf. These parameters essentially control overfitting. In BT algorithms, three other parameters will be considered, linked to the boosting process: the learning rate involved in the update of the output, the number of trees and the subsample share used for each tree that means the fraction of observations to be selected for each tree.

3.3 Experiment plan

3.3.1 Data parameters

The influence of the following parameters is explored in what follows.

- **upscaling ratio.** This is the ratio of the LR grid size to the HR grid size. In [8], numerical experiments are reported with HR and LR spatial resolutions of 62.5 cm and 50 m respectively, which gives an upscaling ratio of 80. For all the simulations in [8], the upscaling ratio was fixed. In the present work, the sensitivity of the downscaling method to the upscaling ratio is explored for a better assessment of the relative performance of the various approaches. In what follows, all methods are tested systematically for upscaling ratios of 5, 20 and 80 (see Table 1).

Symbol	Meaning	Numerical value
L	Domain length	100 m
Δx_{HR}	HR cell size	0.125 m
Δx_{LR}	LR cell size	0.625 m, 5 m, 10 m

Table 1: BVPs : model parameters.

- **variables.** These are defined as the data supplied to the statistical model. They are derived from the variables of the shallow water model. In this work, the following three variables are tested: (i) the pair (h, q) formed by the water depth and the unit discharge, (ii) the square root $\sqrt{h}$ of the water depth, and (iii) the water depth h alone. In option (i), using the full set of flow variables is deemed to maximise the amount of information on the flow state, possibly allowing for an increased accuracy of the downscaling process. In option (ii), the square root of h is proposed as a variable because a number of initial value problems for the shallow water equations have analytical solutions where $\sqrt{h}$ is linear with respect to the space coordinate. Option (iii) is the most obvious one with h, one of the primary variables in the flow equations, being used without any transformation.

3.3.2 Model parameters

Table 2 shows the hyperparameters used for the ANN. Three different structures (1, 2 or 3 hidden layers) were investigated. For each of these structures, two different values were tested for the number of neurons per layer. For each of these 6 combinations (see Table 2), three different values were tested for the number of epochs in the training process. This makes 18 different configurations in total. The batch size was kept constant, equal to 32, for all sets.

Hyperparameters	Numerical values
Number of epochs	50, 150 or 500
Batch size	32
Number of neurons (1-layer configuration)	100 or 500
Number of neurons (2-layer configuration)	(50, 50) or (100,100)
Number of neurons (3-layer configuration)	(50, 50, 50) or (75, 75, 75)

Table 2: ANN hyperparameters. The number of epochs is the number of times the entire data set is used in the training process.

The hyperparameters for BT are provided in Table 3. The learning rate, the number of trees and the subsample share used for each tree are specific to the boosting procedure whereas the maximum depth (global parameter for all trees) and the minimum samples per leaf are specific to the use of random trees. Two values are considered for the minimum samples per leaf. The unit value corresponds to no constraint.

Hyperparameter	Numerical values
Learning rate	0.1
Maximum depth	2 or 4
Minimum samples per leaf	1 sample, 2% of set
Number of trees	7 , 20 or 50
Subsample share used for each tree	50%

Table 3: BT hyperparameters.

4 Results

4.1 Experiment overview

In this section, the simulation results of the experiments presented in Table 4 are presented. The objective is to evaluate the accuracy and robustness of both ML models (artificial neural networks (ANN) and boosted trees (BT)) thanks to various training, validation and testing datasets. Each of the subsections below corresponds to one single experiment (depicted in Table 4), for which the influence of both data and model parameters (see definition in Section 3.3 above) is analysed.

Experiment	Training		Validation		Test	
	h_1 (m)	sample size	h_1 (m)	sample size	h_1 (m)	sample size
1	$0.7, 0.9$	880	$0.7, 0.9$	222	0.8	551
2	$0.7, 0.9$	1102	$0.75, 0.85$	1102	0.8	551
3	$0.7, 0.8$	880	$0.7, 0.8$	222	0.9	551
4	$0.7, 0.85$	1102	$0.75, 0.8$	1102	0.9	551

Table 4: Experiment plan. $h_0 = 1$ m for all simulations.

In Experiment 1, the validation and the training sets are identical. The test value $h_1 = 0.8$m lies within the training and validation sets. In Experiment 2, the validation set is included within the range of the training set, while the test set is unchanged compared to Experiment 1. Experiment 3 is the same as Experiment 1, except that the test set lies outside the range of the former two. Experiment 4 is the same as Experiment 2, except that the test set lies outside the range of the training and validation sets.
The rationale for these experiments is the following. Comparing the tests in Experiments 1 and 2 gives an indication on a possible drop in the performance of the method when the validation set is narrower than the training set. Comparing Experiments 1 and 3 gives an indication on the behaviour of the method when the test set does not lie within the training/validation set. Comparing Experiments 2 and 4 brings information about the influence of a reduced validation interval when the test set lies outside the training/validation intervals.

4.2 Experiment 1

4.2.1 Influence of data parameters

Tables 5 and 6 show the best test performance obtained for the ANN and BT approaches respectively. Two CPU times (T_1, T_2) are provided: T_1 is the time needed to train the ML approach (ANN or BT)

against the training + validation sets, while T_2 is the CPU needed when the approach is run against the test set. These two CPU times are of equal interest in view of operational applications: T_1 gives an idea of the computational effort needed to train the method again after e.g. enriching a data set with new HR-LR observations, while T_2 indicates how fast the method can infer the HR field from the LR field directly, without any further training. For instance, training the BT for UR = 5 against the (h, q) variables requires approximately 30 minutes (see Table 6). For real-time flood crisis management, this is not compatible with online BT training, while online use of an offline trained BT is.
In addition to tables 5 and 6, the mean squared error (MSE) for each of the set of reconstructed variables (both for ANN and BT) is plotted as a function of the upscaling ratio in Figure 4.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	1.2×10^{-5}	4.4×10^{-5}	1.7×10^{-5}
	CPU time (s)	(170, 0.38)	(83, 0.23)	(63, 0.24)
	Param	[500, [100]]	[500, [100]]	[500, [100]]
20	MSE (m^2)	1.7×10^{-5}	3.9×10^{-5}	1.3×10^{-5}
	CPU time (s)	(140, 0.15)	(83, 0.23)	(66, 0.27)
	Param	[500, [100, 100]]	[500, [100]]	[500, [100]]
80	MSE (m^2)	7.4×10^{-5}	7.5×10^{-5}	5.5×10^{-5}
	CPU time	(57, 0.14)	(130, 0.067)	(69, 0.32)
	Param	[500, [75, 75, 75]]	[500, [500]]	[500, [100, 100]]

Table 5: Experiment 1 - ANN best performance. UR: Upscaling Ratio. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, [b_i]]$, with a the number of epochs, b_i the number of neurons in Layer i.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	2.1×10^{-6}	3.9×10^{-7}	$4.3e \times 10^{-7}$
	CPU time (s)	$(1.8 \times 10^{3}, 8.8 \times 10^{-1})$	$(3.8e \times 10^{2}, 3.6 \times 10^{-1})$	$(4.2 \times 10^{2}, 3.7 \times 10^{-1})$
	Param	[4, 1.0, 50]	[4, 1.0, 50]	[4, 1.0, 50]
20	MSE (m^2)	$4. \times 10^{-6}$	$1. \times 10^{-6}$	$8. \times 10^{-7}$
	CPU time (s)	$(4.4 \times 10^{2}, 6.7 \times 10^{-1})$	$(1.3 \times 10^{2}, 3.1 \times 10^{-1})$	$(1.3 \times 10^{2}, 3.1 \times 10^{-1})$
	Param	[4, 1.0, 50]	[4, 1.0, 50]	[4, 1.0, 50]
80	MSE (m^2)	1.1×10^{-5}	1.1×10^{-5}	9.4×10^{-6}
	CPU time (s)	$(1.5 \times 10^{2}, 6.3 \times 10^{-1})$	$(3.8e \times 10^{1}, 2.9 \times 10^{-1})$	$(4.5 \times 10^{1}, 3.8 \times 10^{-1})$
	Param	[4, 1.0, 50]	[4, 0.02, 50]	[2, 1.0, 50]

Table 6: Experiment 1 - BT best performance. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, b, c]$, with a the maximum depth, b the minimum fraction of samples per leaf, c the number of trees.

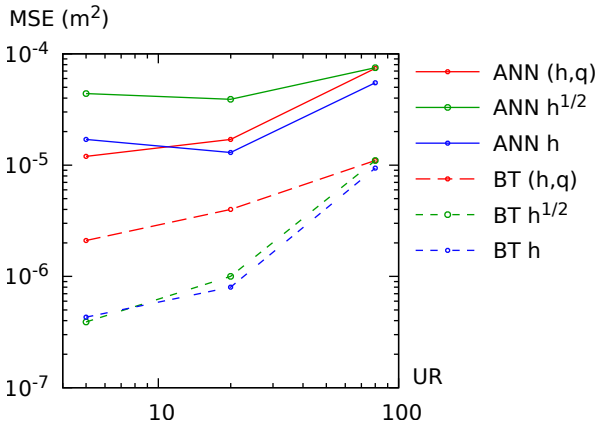

Figure 4: Experiment 1 - ANN and BT best Mean Squared Error (MSE) as a function of the Upscaling Ratio (UR) for the various reconstructed variables.

Both the ANN and BT perform better in reconstructing h than in reconstructing $\sqrt{h}$ and (h, q). The smaller the upscaling ration, the better the results. Overall, BT provide better scores than ANN, with an MSE improved by a factor 10 to 100. However, this is achieved at the expense of CPU training costs 5 to 10 times as large.
Quite expectedly, the ANN perfoms best for the largest number of epochs (500). The complexity of the ANN grows with the upscaling ratio.
The optimal hyperparameters for the BT model vary only slightly. In particular, there is no constraint on the minimal fraction of samples per leaf and a maximum number of trees (50).

4.2.2 Influence of model parameters

In the present section, the performance of each of the two methods is analysed. For the sake of conciseness, only the reconstructed h and the intermediate upscaling ratio UR $= 20$ are considered. The results for ANN ans BT are presented in Tables 7 and 8 respectively.
For each given hyperparameter value, the MSE and CPU times provided in the tables are average values, computed from all possible combinations of the remaining parameters. Consequently, the figures in Tables 7 and 8 can be compared to those presented in the $(h, 20)$ box in Tables 5 and 6 respectively. We recall that Tables 5 and 6 provide the best score (for each single set of parameters), which is a different type of information compared to Tables 7 and 8.

hyperparameter	Value	MSE (m^2)	Training time (s)	Evaluation time (ms)
Epochs	50	3.0×10^{-4}	10.9	0.23
	150	7.2×10^{-5}	24.7	0.24
	500	3.0×10^{-5}	80.6	0.16
Layer structure	[100]	1.1×10^{-4}	31.3	0.20
	[500]	6.8×10^{-5}	64.1	0.18
	[50, 50]	2.0×10^{-4}	38.7	0.25
	[100, 100]	1.1×10^{-4}	32.9	0.16
	[50, 50, 50]	2.0×10^{-4}	38.2	0.23
	[75, 75, 75]	1.1×10^{-4}	27.1	0.22

Table 7: Experiment 1 - ANN performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

hyperparameter	Value	MSE (m^2)	Training time	Evaluation time
Maximum depth	2	4.9×10^{-4}	47.4	0.33
	4	4.7×10^{-4}	60.5	0.30
Minimum samples per leaf	1	4.7×10^{-4}	58.8	0.33
	2%	4.8×10^{-4}	49.2	0.31
Number of trees	7	1.3×10^{-3}	17.3	0.30
	20	9.2×10^{-5}	43.3	0.31
	50	2.6×10^{-6}	101	0.35

Table 8: Experiment 1 - BT performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

The ANN score increases with the number of epochs (as noticed in Subsection 4.2.1), as does the training time. The evaluation time is stable.
The best performing network structure is obtained for a single layer, a large number of neurons yielding a smaller MSE at the expense of a larger training time.

4.3 Experiment 2

4.3.1 Influence of data parameters

The best performing ANN and BT models are presented in Tables 9 and 10 respectively. Figure 5 allows the best performance of both model types to be compared for the three upscaling ratios.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	5.7×10^{-5}	1.8×10^{-5}	3.8×10^{-6}
	CPU time (s)	(3.6×10^{1}, 4.0×10^{-1})	($1.8e \times 10^{2}$, 1.3×10^{-1})	(1.8×10^{2}, 1.3×10^{-1})
	Param	[150, [100]]	[500, [500]]	[500, [500]]
20	MSE (m^2)	$2. \times 10^{-5}$	1.7×10^{-5}	6.7×10^{-6}
	CPU time (s)	(1.1×10^{2}, 4.4×10^{-1})	($6.9 \times 10^{+1}$, 4.9×10^{-1})	(7.9×10^{1}, 2.7×10^{-1})
	Param	[500, [100]]	[500, [100]]	[500, [100]]
80	MSE (m^2)	1.8×10^{-4}	6.7×10^{-5}	4.1×10^{-5}
	CPU time (s)	(1.4×10^{2}, 3.9×10^{-1})	(6.6×10^{1}, 1.9×10^{-1})	(6.4×10^{1}, 1.7×10^{-1})
	Param	[500, [100]]	[500, [75, 75, 75]]	[500, [75, 75, 75]]

Table 9: Experiment 2 - ANN best performance. UR: Upscaling Ratio. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, [b_i]]$, with a the number of epochs, b_i the number of neurons in Layer i.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	6.5×10^{-3}	1.6×10^{-3}	1.6×10^{-3}
	CPU time	(2.3×10^{3}, 3.6)	(5.3×10^{2}, 1.2)	(5.3×10^{2}, 1.2)
	Param	[4, 1.0, 50]	[4, 1.0, 50]	[4, 1.0, 50]
20	MSE (m^2)	6.0×10^{-3}	1.3×10^{-3}	1.4×10^{-3}
	CPU time	(5.5×10^{2}, 2.0)	(1.9×10^{2}, 8.9×10^{-1})	(1.2×10^{2}, 8.4×10^{-1})
	Param	[4, 1.0, 50]	[4, 1.0, 50]	[4, 0.02, 50]
80	MSE (m^2)	5.5×10^{-3}	1.6×10^{-3}	1.6×10^{-3}
	CPU time	(1.5×10^{2}, 1.6)	(4.4×10^{1}, 7.8×10^{-1})	(5.2×10^{1}, 7.7×10^{-1})
	Param	[4, 0.02, 50]	[4, 0.02, 50]	[4, 0.02, 50]

Table 10: Experiment 2 - BT best performance. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, b, c]$, with a the maximum depth, b the minimum fraction of samples per leaf, c the number of trees.

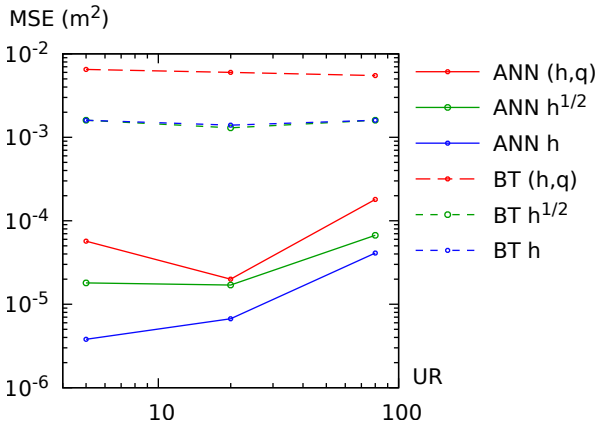

Figure 5: Experiment 2 - ANN and BT best Mean Squared Error (MSE) as a function of the Upscaling Ratio (UR) for the various reconstructed variables.

This experiment yields opposite benchmarking results to those of Experiment 1 in that ANN consistently perform better than BT, regardless of the value of UR and the reconstructed variable. Besides, while the accuracy of the ANN tends to improve when smaller upscaling ratios are used, the best MSE for BT remains strikingly independent of UR. Moreover, the MSE for BT is between 10^{-3}m^2 and 5×10^{-3} m^2, which gives an L-2 norm between 3 cm and 7 cm for the BT modelling error. This is $10\tilde{\%}$ to $24\tilde{\%}$ of the amplitude of the strongest wave transient ($|h_0 - h_1| = 30$ cm at most), which can be regarded as a large relative modelling error.

4.3.2 Influence of model hyperparameters

hyperparameter	Value	MSE (m^2)	Training time	Evaluation time
Epochs	50	2.6×10^{-4}	11.4	0.32
	150	5.3×10^{-5}	32.8	0.27
	500	2.4×10^{-5}	92.3	0.27
Layer Structure	[100]	1.1×10^{-4}	37.7	0.29
	[500]	4.1×10^{-5}	76.0	0.20
	[50, 50]	1.4×10^{-4}	44.7	0.31
	[100, 100]	1.2×10^{-4}	46.6	0.38
	[50, 50, 50]	1.6×10^{-4}	33.5	0.22
	[75, 75, 75]	1.1×10^{-4}	34.5	0.31

Table 11: Experiment 2 - ANN performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

hyperparameter	Value	MSE (m^2)	Training time	Evaluation time
Maximum depth	2	2×10^{-3}	57.6	0.67
	4	2×10^{-3}	75.2	0.67
Minimum samples per leaf	1	2×10^{-3}	72.4	0.70
	2%	2×10^{-3}	60.4	0.64
Number of trees	7	2×10^{-3}	21.3	0.49
	20	2×10^{-3}	53.6	0.62
	50	2×10^{-3}	124.3	0.89

Table 12: Experiment 2 - BT performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

Quite expectedly, the performance of the ANN increases with the number of epochs. The best performance is obtained with a single layer containing 500 neurons. All the BT tested exhibit similar (large) MSE values. The training time appears to be roughly proportional to the number of trees and seems to follow a sublinear trend with respect to the maximum depth. The evaluation CPU time has the same order of magnitude for both methods.

4.4 Experiment 3

4.4.1 Influence of data parameters

It is recalled that in Experiment 3, the training and validation are carried out using the same range for h_1, but that the test set lies outside the range of the training-validation interval. The best performance of the ANN and BT are presetned in Tables 13 and 14 respectively.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	1.5×10^{-5}	7.1×10^{-5}	1.1×10^{-5}
	CPU time (s)	$(1.4e \times 10^2 1.6 \times 10^{-1})$	$(5.6 \times 10^1, 1.3 \times 10^{-1})$	$(6.2 \times 10^1, 2.4 \times 10^{-1})$
	Param	[500, [50, 50, 50]]	[500, [75, 75, 75]]	[500, [100]]
20	MSE (m^2)	1.3×10^{-5}	2.6×10^{-5}	9.8×10^{-6}
	CPU time (s)	$(7.9 \times 10^1, 1.7 \times 10^{-1})$	$(5.5 \times 10^1, 1.7 \times 10^{-1})$	$(5.9 \times 10^1, 3.2 \times 10^{-1})$
	Param	[500, [75, 75, 75]]	[500, [75, 75, 75]]	[500, [100]]
80	MSE (m^2)	3.76×10^{-5}	5.99×10^{-5}	4.45×10^{-5}
	CPU time (s)	$(8.0 \times 10^1, 1.6 \times 10^{-1})$	$(8.3 \times 10^1, 3.1 \times 10^{-1})$	$(8.3 \times 10^1, 7.7 \times 10^{-2})$
	Param	[500, [50, 50, 50]]	[500, [100, 100]]	[500, [75, 75, 75]]

Table 13: Experiment 3 - ANN best performance. UR: Upscaling Ratio. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, [b_i]]$, with a the number of epochs, b_i the number of neurons in Layer i.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	1.9×10^{-6}	4.9×10^{-7}	4.1×10^{-7}
	CPU time (s)	$(2.7 \times 10^{3}, 1.5)$	$(4.6 \times 10^{2}, 3.7 \times 10^{-1})$	$(3.9 \times 10^{2}, 3.8 \times 10^{-1})$
	Param	[4, 1.0, 50]	[4, 1.0, 50]	[4, 1.0, 50]
20	MSE (m^2)	3.4×10^{-6}	7.8×10^{-7}	6.9×10^{-7}
	CPU time (s)	$(7.2 \times 10^{2}, 1.3)$	$(1.4 \times 10^{2}, 3.2 \times 10^{-1})$	$(1.4 \times 10^{2}, 3.2 \times 10^{-1})$
	Param	[4, 1.0, 50]	[4, 1.0, 50]	[4, 1.0, 50]
80	MSE (m^2)	1.78×10^{-5}	7.34×10^{-6}	7.53×10^{-6}
	CPU time (s)	$(1.7 \times 10^{2}, 6.7 \times 10^{-1})$	$(4.5 \times 10^{1}, 3.8 \times 10^{-1})$	$(4.6 \times 10^{1}, 3.8 \times 10^{-1})$
	Param	[4, 1.0, 50]	[2, 1.0, 50]	[2, 1.0, 50]

Table 14: Experiment 3 - BT best performance. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, b, c]$, with a the maximum depth, b the minimum fraction of samples per leaf, c the number of trees.

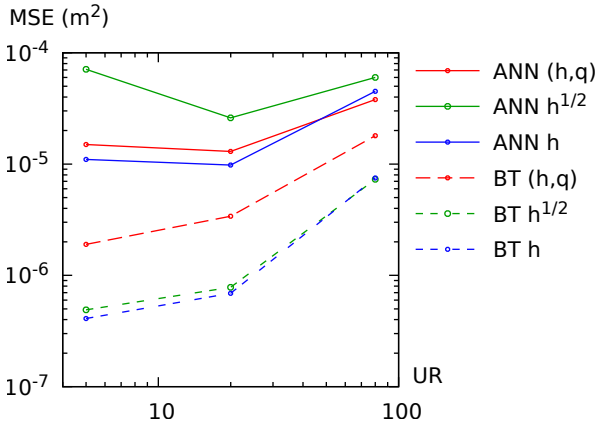

Figure 6: Experiment 3 -ANN and BT best Mean Squared Error (MSE) as a function of the Upscaling Ratio (UR) for the various reconstructed variables.

As in Experiment 1, the best BT performs better than the best ANN for all three upscaling ratios. While the ANN models seem to perform optimally for intermediate values of the upscaling ratio, the accuracy of the best BT exhibits a monotone behaviour with respect to UR. For BT, reconstructing h or $\sqrt{h}$ consistently yields significantly better results than reconstructing (h, q). For ANN, reconstructing $\sqrt{h}$ yields poorer performance than reconstructing h or (h, q). However, the reconstruction options (h, $\sqrt{h}$ and (h, q)) tend to give similar performance as the upscaling ratio increases.

4.4.2 Influence of model parameters

Tables 15 and 16 provide the performance of the ANN and BT respectively for various values of the hyperparameters. As in the previous sections, only the reconstruction of h is considered, for an upscaling

ratio UR = 20.

hyperparameter	Value	MSE (m^2)	Training time	Evaluation time
Epochs	50	3.4×10^{-4}	10.3	0.21
	150	7.5×10^{-5}	28.1	0.31
	500	1.5×10^{-5}	71.5	0.20
Layers structure	[100]	1.3×10^{-4}	29.4	0.24
	[500]	5.9×10^{-5}	62.7	0.18
	[50, 50]	1.771×10^{-4}	37.5	0.22
	[100, 100]	1.3×10^{-4}	31.3	0.26
	[50, 50, 50]	2.1×10^{-4}	28.2	0.31
	[75, 75, 75]	1.5×10^{-4}	30.7	0.23

Table 15: Experiment 3 - ANN performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

hyperparameter	Value	MSE (m^2)	Training time	Evaluation time
Maximum depth	2	4.8×10^{-4}	50.1	0.34
	4	4.6×10^{-4}	66.1	0.32
Minimum samples per leaf	1	4.7×10^{-4}	63.1	0.33
	2%	4.8×10^{-4}	53.1	0.33
Number of trees	7	1.3×10^{-3}	18.2	0.30
	20	9.3×10^{-5}	50.1	0.34
	50	3.8×10^{-6}	106	0.36

Table 16: Experiment 3 - BT performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

As in the previous two experiments, the performance of the ANN increases with the number of epochs. The optimal structure is a single layer containing 500 neurons.
The performance of the BT depends mostly on the number of trees, while the maximum depth and minimum samples per leaf have a marginal influence on the results.

4.5 Experiment 4

4.5.1 Influence of data parameters

In this experiment, the validation set is narrower than the training set, and the test set lies outside the training interval. The performance of the best ANN and BT models is summarized in Tables 17 and 18 respectively. The variations of the MSE with the upscaling ratio are illustrated in Figure 7.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	9.3×10^{-5}	8.4×10^{-5}	1.8×10^{-5}
	CPU time (s)	$(1.4 \times 10^{2}, 1.0)$	$(1.4 \times 10^{2}, 8.1 \times 10^{-1})$	$(1.0 \times 10^{2}, 9.4 \times 10^{-1})$
	Param	[150, [500]]	[500, [100, 100]]	[500, [100]]
20	MSE (m^2)	7.6×10^{-5}	5.6×10^{-5}	2.1×10^{-5}
	CPU time (s)	$(3.3 \times 10^{2}, 1.6 \times 10^{-1})$	$(1.9 \times 10^{2}, 2.2 \times 10^{-1})$	$(9.9 \times 10^{1}, 6.1 \times 10^{-1})$
	Param	[500, [500]]	[500, [500]]	[500, [50, 50]]
80	MSE (m^2)	2.8×10^{-4}	1.0×10^{-4}	1.0×10^{-4}
	CPU time (s)	$(1.1 \times 10^{2}, 4.9 \times 10^{-1})$	$(9.2 \times 10^{1}, 4.4 \times 10^{-1})$	$(5.1 \times 10^{1}, 1.2 \times 10^{-1})$
	Param	[500, [100]]	[500, [100]]	[500, [75, 75, 75]]

Table 17: Experiment 4 - ANN best performance. UR: Upscaling Ratio. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, [b_i]]$, with a the number of epochs, b_i the number of neurons in Layer i.

UR	Best performance	Reconstructed variable		
		(h, q)	$\sqrt{h}$	h
5	MSE (m^2)	4.4×10^{-3}	1.6×10^{-3}	1.6×10^{-3}
	CPU time (s)	$(2.0 \times 10^{3}, 6.0)$	$(4.6 \times 10^{2}, 1.7)$	$(4.2 \times 10^{2}, 1.5)$
	Param	[2, 1.0, 50]	[2, 0.02, 50]	[2, 0.02, 50]
20	MSE (m^2)	4.4×10^{-3}	1.5×10^{-3}	1.6×10^{-3}
	CPU time (s)	$(5.6 \times 10^{2}, 3.1)$	$(1.5 \times 10^{2}, 1.2)$	$(1.3 \times 10^{2}, 1.0)$
	Param	[2, 1.0, 50]	[2, 1.0, 50]	[2, 0.02, 50]
80	MSE (m^2)	4.4×10^{-3}	1.6×10^{-3}	1.6×10^{-3}
	CPU time (s)	$(2.0 \times 10^{2}, 1.9)$	$(6.8 \times 10^{1}, 8.9 \times 10^{-1})$	$(5.8 \times 10^{1}, 9.5 \times 10^{-1})$
	Param	[2, 0.02, 50]	[2, 1.0, 50]	[2, 0.02, 50]

Table 18: Experiment 4 - BT best performance. CPU time format: (T_1, T_2), with T_1 the training time, T_2 the evaluation time. Hyperparameter format (Param): $[a, b, c]$, with a the maximum depth, b the minimum fraction of samples per leaf, c the number of trees.

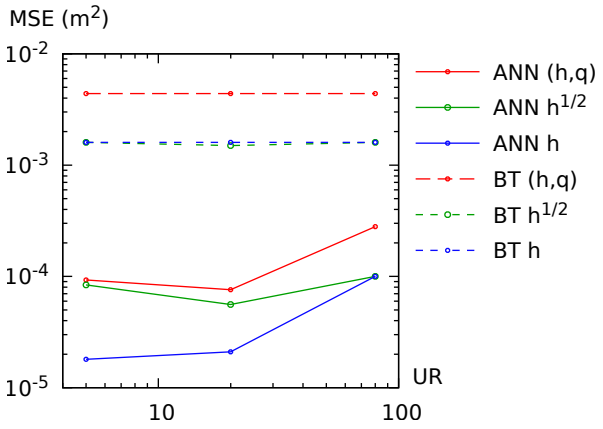

Figure 7: Experiment 4 - ANN and BT best Mean Squared Error (MSE) as a function of the Upscaling Ratio (UR) for the various reconstructed variables.

As in Experiment 2, the best ANN performs significantly better than the best BT, by at least one order of magnitude. Reconstructing (h, q) yields the poorest performance for both approaches. For ANN, h is better reconstructed than $\sqrt{h}$ for smaller UR values, while these two variables are reconstructed with the same (low) degree of accuracy by BT. The performance of the best BT is almost independent of UR.

4.5.2 Influence of model parameters

The sensitivity of model performance is presented in Tables 19 and 20

hyperparameter	Value	MSE (m^2)	Training time	Evaluation time
Epochs	50	5.8×10^{-4}	14.1	0.61
	150	1.3×10^{-4}	41.5	0.61
	500	2.9×10^{-5}	118	0.32
Layers structure	[100]	2.2×10^{-4}	50.7	0.59
	[500]	7.8×10^{-5}	100	0.45
	[50, 50]	3.2×10^{-4}	47.4	0.51
	[100, 100]	2.5×10^{-4}	64.0	0.52
	[50, 50, 50]	3.9×10^{-4}	42.3	0.59
	[75, 75, 75]	2.2×10^{-4}	42.6	0.44

Table 19: Experiment 4 - ANN performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

hyperparameter		MSE (m^2)	Training time	Evaluation time
Maximum depth	2	2×10^{-3}	66.4	0.76
	4	2×10^{-3}	87.2	0.79
Minimum samples per leaf	1	2×10^{-3}	82.4	0.79
	2%	2×10^{-3}	71.2	0.76
	7.00	4×10^{-3}	24.3	0.52
Number of trees	20	2×10^{-3}	64.1	0.78
	50	2×10^{-3}	142	1.02

Table 20: Experiment 4 - BT performance for various hyperparameter sets. Reconstructed variable: h. UR: 20.

Table 19 confirms the trend observed in the previous three experiments for ANN, i.e. an increasing accuracy with the number of epochs and a single-layered optimal structure with 500 neurons. Table 20 confirms that the BT performance is almost insensitive to the values of the hyperparameters.

5 Conclusions

In this paper, we have applied two downscaling approaches to the well-known shallow water equations, that are widely used in hydraulic engineering applications. The numerical solution of the shallow water equations is computed at various space resolutions. AI algorithms (ANNs and BTs) are applied to these low resolution fields in order to obtain a high resolution solution, considered as the reference. In all experiments, the AI methods are trained and validated on low resolution simulations. The corresponding loss is evaluated thanks to a pre-computed, high resolution reference solution. A test set is used to evaluate the performance of each of the methods, with various parameter settings.

With 4 different numerical experiments in which datasets are modified through the flow boundary conditions, we observe that neural networks provide very stable and efficient results. Unsurprisingly, the associated score and computational cost increase with the number of epochs, as well as with the number of neurons. We also notice that for the same number of neurons, a single-layer structure performs better than a more complex structure with 2 or 3 layers. This confirms the results obtained in [8].
When the validation set is identical to the traning set, BTs perform significantly better than ANNs, with an error 1 to 2 orders of magnitude as small and a computational cost only 5 to 10 times as large. However, their performance is strikingly degraded when the validation set is not identical to the training set. In such a case, the modelling error is at least one order of magnitude as large as that of ANNs.

Acknowledgement

The authors are grateful to the Technological Research Institute Saint-Exupéry[1], who partially supported this work, as well as the french national programme LEFE/INSU.

[1]See `https://www.irt-saintexupery.com/fr/`

References

[1] J.L. Auriault. Heterogeneous medium. is an equivalent macroscopic description possible? *International Journal of Engineering Science*, 29:785–795, 1995.

[2] J.L. Auriault and P. Adler. Taylor dispersion in porous media: analysis by multiple scale expansions. *Advances in Water Resources*, 18:217–226, 1995.

[3] A. Bensoussan, J.L. Lions, and G.C. Papanicolaou. *Asymptotic analysis for periodic structures.* North-Holland, Amsterdam, 1980.

[4] Bernardin, Frédéric, Bossy, Mireille, Chauvin, Claire, Jabir, Jean-François, and Rousseau, Antoine. Stochastic lagrangian method for downscaling problems in computational fluid dynamics. *ESAIM: M2AN*, 44(5):885–920, 2010.

[5] C. M. Bishop. *Pattern Recognition and Machine Learning.* Springer, 2006.

[6] M. Bruwier, P. Archambeau, S. Erpicum, M. Pirotton, and B. Dewals. Shallow-water models with anisotropic porosity and merging for flood modelling on cartesian grids. *Journal of Hydrology*, 554:693–709, 2017.

[7] A.J Cannon and P.H. Whitfield. Downscaling recent streamflow conditions in british columbia, canada using ensemble neural network models. *Journal of Hydrology*, 259:136–151, 2002.

[8] J. Carreau and V. Guinot. A PCA spatial pattern based artificial neural network downscaling model for urban flood hazard assessment. *Advances in Water Resources*, 147:103821, January 2021.

[9] A. Chen, B. Evans, S. Djordjevic, and D.A. Savic. A coarse-grid approach to represent building blockage effects in 2d urban flood modelling. *Journal of Hydrology*, 426-427:1–16, 2012.

[10] A. Defina. Two-dimensional shallow flow equations for partially dry areas. *Water Resour Res*, 36:3251–3264, 2000.

[11] Y.B. Dibike and P.. Coulibaly. Temporal neural networks for downscaling climate variability and extremes. *Neural Networks*, 19:135–144, 2006.

[12] H.I. Ene and E. Sanchez-Palencia. *Non-homogeneous media and vibration theory.* Springer, Berlin, 1980.

[13] C.L. Farmer. Upscaling: a review. *International Journal for Numerical Methods in Fluids*, 40:63–78, 2002.

[14] Niels Fraehr, Quan J. Wang, Wenyan Wu, and Rory Nathan. Upskilling low-fidelity hydrodynamic models of flood inundation through spatial analysis and gaussian process learning. *Water Resources Research*, 58(8), August 2022.

[15] Y. Freund. Boosting a weak learning algorithm by majority. *Information and Computation*, 121(2):256–285, September 1995. Also appeared in COLT90.

[16] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *Proceedings of the 13th international conference on machine learning*, pages 148–156. Morgan Kaufmann, 1996.

[17] Yoav Freund and Robert E. Schapire. A short introduction to boosting. In *In Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1401–1406. Morgan Kaufmann, 1999.

[18] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.

[19] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5):1189–1232, 2001.

[20] S.K. Godunov. A difference method for numerical calculation of discontinuous solutions of the equations of hydrodynamics. *Mat. Sb.*, 47(89):271–306, 1959.

[21] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`.

[22] V. Guinot. Multiple porosity shallow water models for macroscopic modelling of urban floods. *Advances in Water Resources*, 37:40–72, 2012.

[23] V. Guinot, Sanders B.F., and Schubert J.E. Dual integral porosity shallow water model for urban flood modelling. *Advances in Water Resources*, 103:16–31, 2017.

[24] V. Guinot, Delenne C., and S. Soares-Frazão. Urban dambreak experiments - riverflow paper. In *RiverFlow 2018 ibternational Conference*, 2018.

[25] V. Guinot and S. Soares-Frazão. Flux and source term discretization in two-dimensional shallow water models with porosity on unstructured grids. *International Journal for Numerical Methods in Fluids*, 50:309–345, 2006.

[26] Amiram Harten, Peter D. Lax, and Bram van Leer. On upstream differencing and godunov-type schemes for hyperbolic conservation laws. *SIAM Review*, 25(1):35–61, January 1983.

[27] J.M. Hervouët, R. Samie, and B. Moreau. Modelling urban areas in dam-break floodwave numerical simulations. In *Proceedings of the international seminar and workshop on rescue actions based on dambreak flow analysis, Seinâjoki, Finland, 1–6 October 2000*, 2000.

[28] P. Indelman and G. Dagan. Upscaling of permeability of anisotropic heterogeneous formations. 1. the general framework. *Water Resources Research*, 29:917–923, 1993.

[29] B. Kim, B.F. Sanders, J.S. Famiglietti, and V. Guinot. Urban flood modeling with porous shallow-water equations: A case study of model errors in the presence of anisotropic porosity. *Journal of Hydrology*, 523:680–692, 2015.

[30] P. D. Lax. Hyperbolic systems of conservation laws II. *Communications on Pure and Applied Mathematics*, 10(4):537–566, 1957.

[31] Yann Lecun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature Cell Biology*, 521(7553):436–444, May 2015.

[32] A. Luke, B. F. Sanders, K. A. Goodrich, D. L. Feldman, D. Boudreau, A. Eguiarte, K. Serrano, A. Reyes, J. E. Schubert, A. AghaKouchak, V. Basolo, and R. A. Matthew. Going beyond the flood insurance rate map: insights from flood hazard map co-production. *Natural Hazards and Earth System Sciences*, 18(4):1097–1120, 2018.

[33] P. Vaittinada Ayar M. Vrac. Influence of bias correcting predictors on statistical downscaling models. *Journal of Applied Meteorology and Climatology*, 56:5–26, 2016.

[34] P. Renard and G. De Marsily. Calculating equivalent permeability: a review. *Advances in Water Resources*, 20:253–278, 1997.

[35] B.F. Sanders, J.E. Schubert, and H.A. Gallegos. Integral formulation of shallow water models with anisotropic porosity for urban flood modelling. *Journal of Hydrology*, 362:19–38, 2008.

[36] Brett F. Sanders and Jochen E. Schubert. PRIMo: Parallel raster inundation model. *Advances in Water Resources*, 126:79–95, April 2019.

[37] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, Jan 2015.

[38] Jochen E. Schubert and Brett F. Sanders. Building treatments for urban flood inundation models and implications for predictive skill and modeling efficiency. *Advances in Water Resources*, 41:49–64, June 2012.

[39] Joao G. Caldas Steinstraesser, Vincent Guinot, and Antoine Rousseau. Modified parareal method for solving the two-dimensional nonlinear shallow water equations using finite volumes. *The SMAI journal of computational mathematics*, 7:159–184, 2021.

[40] M. Vrac T. Salameh, P. Drobinski and P. Naveau. Statistical downscaling of near surface wind field over complex terrain in southern france. *Meteorology and Atmospheric Physics*, 103:253–265, 2009.

[41] Daniele P. Viero. Modelling urban floods using a finite element staggered scheme with an anisotropic dual porosity model. *Journal of Hydrology*, 568:247–259, January 2019.

[42] Daniele Pietro Viero and Mohammad Valipour. Modeling anisotropy in free-surface overland and shallow inundation flows. *Advances in Water Resources*, 104:1–14, June 2017.

[43] I. Özgen, J. Zhao, D. Liang, and R. Hinkelmann. Urban flood modeling using shallow water eqations with depth-dependent anisotropic porosity. *Journal of Hydrology*, 541:1165–1184, 2016.