Supplementary Material

Critical factors for the use of machine learning to predict lake surface water temperature

Azadeh Yousefi, Marco Toffolon

Department of Civil, Environmental and Mechanical Engineering, University of Trento, Italy

This Supplementary Material contains the following sections.

1. The list of the acronyms used in the text.
2. The thermal behaviour simulated by the one-dimensional lake model General Lake Model (GLM) in lakes of different depth.
3. A summary of the literature about modelling stream temperature in rivers.
4. The list of the metrics used to evaluate the performance of machine learning (ML) and their values.
5. The values of the hyperparameters optimized for the examined ML approaches.

1. Acronyms used in the text

Table S1 reports the list of the acronyms used in the main text and in this Supplementary Material.

Table S1. List of acronyms.

ANFIS	Adaptive Neuro-Fuzzy Inference System
ANN	Artificial Neural Network
AP	Air Pressure
AT	Air Temperature
BPNN	Back-Propagation Neural Network
CWT	Continuous Wavelet Transform
DL	Deep Learning
DOY	Day Of the Year
DT	Decision Tree
DWT	Discrete Wavelet Transform
ERT	Extremely Randomized Tree
GA	Genetic Algorithm
GAM	Generalized Additive Model
GEP	Gene Expression Programming, a variant of Genetic Programming
GP	Genetic Programming
KNN	K-Nearest Neighbour
LSTM	Long Short-Term Memory
LSWT	Lake Surface Water Temperature
LWR	Longwave Radiation
ML	Machine Learning
MLPNN	Multi-Layer Perceptron Neural Network
PGRNN	Physics-Guided Recurrent Neural Network
PSO	Particle Swarm Optimization
R	Rainfall
RF	Random Forest
RH	Relative Humidity
RMSE	Root Mean Square Error
RNN	Recurrent Neural Network
RT	Regression Tree
SVM	Support Vector Machine
SVR	Support Vector Regression
SWR	Shortwave Radiation
T	Temperature
WS	Wind Speed
WT	Wavelet Transform

2. Simulated thermal behaviour in lakes of different depth

Fig. S1 shows the temporal evolution of water temperature along the water column in two versions of the synthetic lakes (shallow and deep), as obtained by numerical simulations with the one-dimensional lake model, General Lake Model (GLM). Two subsequent years are shown to highlight the typical seasonal patterns, and who they differ from the shallow to the deep case.

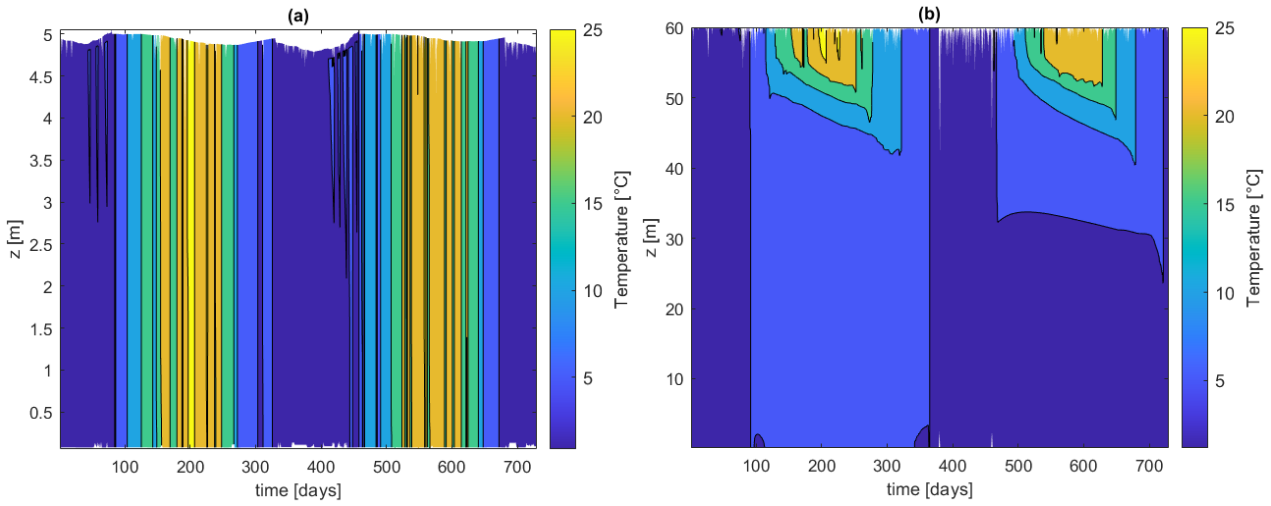

Figure S1. Evolution of water temperature in two lakes with the same hypsometry, but rescaled with different depths (a: maximum depth 5 m; b: maximum depth 60 m), as a function of depth and time (from 01/01/2015 to 31/12/2016), obtained from the physically based model GLM.

3. Modelling stream temperature in rivers

A list of ML applications to predict stream temperature in rivers is reported in Table S2. For the case of rivers, the most influential predictors are AT, discharge, and DOY, while some features seem to influence the performances negatively, such as catchment extents and forest land cover in riparian zones (DeWeber and Wagner, 2014) and low flow rates (Rivers-Moore et al., 2005). We note that the relationship between AT and water temperature tends to be approximately linear in streams (Erickson and Stefan, 2000), similar to the case of shallow lakes.

Table S2. ML algorithms used to predict stream temperature in rivers, with the indication of the most important predictors.

Type of ML algorithm	**Author(s)**	**ML methods**	**The best ML method**	**Predictor(s)**
Linear and/or Non-linear Regression	Mohseni et al. (1998)	Non-linear regression model	Non-linear regression model	Weekly AT
	Mohseni and Stefan (1999)	Linear regression	Linear regression	AT
	Rivers-Moore et al. (2005)	Linear regression, Non-linear regression model (Exponential model), Multiple linear regression model	Multiple linear regression model	AT, streamflow, rainfall, relative humidity (mean daily AT, minimum daily AT and RH are influential)
	Li et al. (2014)	Varying coefficient regression model	Varying coefficient regression model	Maximum AT, time varying coefficient model (DOY)
Logistic Regression	Caissie et al. (2001)	Regression model (second order Markov process), using Fourier and Sine function for pre-processing	Regression model on weekly basis; stochastic for maximum AT	AT
	Grbic' et al. (2013)	Gaussian Process Regression - Linear regression, logistic and stochastic models	Gaussian process regression model	AT, streamflow
	Laanaya et al. (2017)	The generalized additive model (GAM), an extension of the generalized linear model; logistic model; linear regression; residual regression model (long-term annual and short-term)	GAM	AT, average streamflow
Artificial Neural Network (ANN)	Chenard and Caissie (2008)	ANN	ANN	Minimum, maximum and mean AT, DOY, water level
	Sahoo et al. (2009)	Empirical model (BPNN), a statistical model (multiple regression analysis, MRA), chaotic non-linear dynamic algorithms (CNDA)	ANN	SWR, AT
	Wenxian et al. (2010)	BPNN	PNN	Monthly water T
	DeWeber and Wagner (2014)	ANN	ANN	Daily averaged AT, prior 7-day mean AT, network catchment area, predicted mean daily water T (forest land cover at riparian and catchment extents impact negatively)
	Hadzima-Nyarko et al. (2014)	ANN; linear regression model and stochastic model	ANN	Daily average AT
	Napiorkowski et al. (2014)	ANN	ANN	Daily average AT
	Piotrowski et al. (2014)	ANN	ANN	Daily average AT, daily maximum AT, daily streamflow, daily average water T, declination of the sun
	Piotrowski et al. (2015)	MLPNN; ANFIS	ANN	Mean, maximum and minimum daily AT, streamflow, SWR.
	Rabi et al. (2015)	Linear regression, stochastic modelling (non-linear regression); MLPNN	MLPNN	Mean daily AT
	Piotrowski et al. (2016)	ANFIS; MLPNN	ANN	SWR, streamflow, minimum daily AT, mean daily AT, maximum daily AT, sum of the daily averaged AT measured 2 to 6 days before the day.
	Temizyurek and Dadaser-Celik (2018)	ANN	ANN	AT, WS, RH, previous water T

	Zhu et al. (2018)	ANN; GPR; BA-DT	Gaussian Process Regression (GPR), Bootstrap Aggregated Decision Trees (BA-DT)	AT
	Graf et al. (2019)	WT-ANN; linear and non-linear regression; ANN	WT-ANN	Daily water T, daily AT
	Trinh et al. (2019)	Regression models (linear, non-linear and stochastic regression); ANN	ANN	Daily maximum AT
	Zhu and Heddam (2019)	Optimally Pruned Extreme Learning Machine (OPELM); Radial Basis Functions Neural Networks (RBFNN)	OPELM	AT, streamflow, DOY
	Zhu et al. (2019 a and b)	MLPNN; ANFIS	MLPNN	AT, streamflow, DOY
	Zhu et al. (2019)	MLPNN and ANFIS with and without WT	WT-MLPNN and WT-ANFIS	Daily AT, DOY
	Zhu et al. (2019)	Extreme Learning Machine (ELM) - MLPNN and simple multiple linear regression (MLR)	ELM	AT, streamflow, DOY
	Zhu et al. (2019)	MLPNN; Gaussian process regression (GPR); DT; *air2stream*	*air2stream* model outperformed the three ML methods	AT, streamflow, DOY
	Zhu et al. (2019)	MLPNN; ANFIS	MLPNN	AT, streamflow, DOY
	Piotrowski et al. (2020)	ANN	ANN	AT, streamflow, declination of the sun
	Qiu et al. (2020)	BPNN; radial basis function neural network; wavelet neural network; general regression neural network; Elman neural network	BPNN	AT, streamflow, DOY
	Radulescu (2020)	MLPNN	MLPNN	AT
	Zhu et al. (2020)	MLPNN; MLPNN integrated model (WT_MLPNN); non-linear regression model (S-curve); *air2stream*	WT_MLPNN (*air2stream* outperformed the other models)	Daily LSWT and AT
Decision Tree (DT)	Goyal et al. (2012)	Tree algorithms (single conjunctive rule learner, decision table, M5 model tree, and REPTree)	M5 model tree	Specific humidity, geopotential height, meridional (north-south direction) WS, zonal (west-east direction) WS, AT
Support Vector Machines (SVM)	Rehana (2019)	Multiple Linear Regression Model (MLRM); SVR	SVR	AT, streamflow
k-nearest Neighbour algorithm (KNN)	St-Hilaire et al. (2012)	KNN	KNN	Water T from the two previous days and an indicator of seasonality (DOY), daily AT and daily streamflow
Random Forest (RF)	Holthuijzen (2017)	Gradient Boosting Machines (GBM); RF; Spatial Statistical Network (SSN); Generalized Additive Models (GAM); Linear regression	GBM	SWR, summer streamflow, maximum weekly maximum AT, summer mean AT
	Lu and Ma (2020)	Extreme Gradient Boosting (XGBoost), RF	RF	1875 data

4. Metrics used to evaluate ML performance

4.1 Standard metrics

Our analysis of the ML performance is based on the values of the root mean square error (RMSE), which is computed for both the training and test data using the standard definition

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(sim_i - obs_i)^2} \tag{S1}$$

where i is the time index, N is the number of samples, sim_i (simulated) is the LSWT modelled by ML, and obs_i is the observed value (obtained from GLM simulation). Perfect fit is obtained for RMSE = 0.

However, different metrics are also used. In this section, we review the most common ones. First, the Nash-Sutcliff Efficiency index (NSE) is equivalent to RMSE if the variance of the observations, σ_{obs}^2, does not change:

$$NSE = 1 - \frac{\sum_{i=1}^{N}(sim_i - obs_i)^2}{\sum_{i=1}^{N}\left(obs_i - \overline{obs}\right)^2} = 1 - \frac{RMSE^2}{\sigma_{obs}^2} \tag{S2}$$

where $\overline{obs}$ is the mean of the observations. NSE = 1 indicates perfect fit, while suing the mean of the observations as a predictive model would lead to NSE = 0.

Other metrics that are used to compute the performance of models are the mean absolute error (MAE)

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|sim_i - obs_i| \tag{S3}$$

which considers the absolute value of the error instead of the square as in equation (S1), and the mean squared error (MSE)

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(sim_i - obs_i)^2 = RMSE^2 \tag{S4}$$

which is simply the square of RMSE.

Another metric is the Index of Agreement presented by Willmott (1981),

$$IA = 1 - \frac{\sum_{i=1}^{N}(sim_i - obs_i)^2}{\sum_{i=1}^{N}\left(\left|sim_i - \overline{obs}\right| + \left|obs_i - \overline{obs}\right|\right)^2} \tag{S5}$$

which is the ratio of MSE and the potential error. IA is sensitive to extreme values because of the squared differences. The values of IA range between 0 and 1, where 1 represents the perfect prediction and 0 the worst.

The coefficient of determination, or R-squared (R^2), yields the same information as NSE. Usually, it is defined as

$$R^2 = 1 - \frac{\sum_{i=1}^{N}\left(\widehat{obs}_i - obs_i\right)^2}{\sum_{i=1}^{N}\left(obs_i - \overline{obs}\right)^2} \tag{S6}$$

where the only difference is the interpretation of the term $\widehat{obs}_i$, which in this case is the prediction from a statistical model based on the observations, and not from the results of a simulation model as for NSE. We do not report the R^2 values as we prefer to indicate them as NSE.

Finally, we use the mean error (ME), or bias, to compute the average difference between simulated and observed values:

$$ME = \frac{1}{N}\sum_{i=1}^{N}(sim_i - obs_i) = \overline{sim} - \overline{obs} \tag{S7}$$

where $\overline{sim}$ is the average of the simulated LSWT.

4.2 Robustness of multiple runs

The results of the ML models depend on the initial guess of their parameters, which are then suitably calibrated by means of algorithms as GA. In order to test the possible variability of the results due to the random choice of the initial parameters, we decided to run all ML models N_r times, and we assumed $N_r = 20$.

Here we introduce two concepts: the robustness and the accuracy of the ML model. The robustness shows how much the results of the N different simulations are close to each other. In order to have an estimate of the robustness, at each time step i we compute the standard deviation among the N_r runs:

$$\sigma_i = \frac{1}{N_r}\sum_{j=1}^{N_r}\left(sim_{i,j} - \mu_{sim,i}\right)^2 \tag{S8}$$

where j is the index of the run, and $\mu_{sim,i}$ is the mean of the simulated LSWT. Then, we compute the average over the whole record (Equation S8) to obtain the index for the train and test sets separately.

$$\sigma_R = \frac{1}{N}\sum_{i=1}^{N}\sigma_i \tag{S9}$$

The accuracy represents how much the prediction is close to the observed values. We already defined the RMSE in equation (S1) as a measure that can be used for a single run. Here, we define two metrics for the set of N_r runs. The first one is the mean of the RMSE of the single runs,

$$RMSE_A = \frac{1}{N_r}\sum_{j=1}^{N_r} RMSE_j \tag{S10}$$

and $RMSE_B$ is the best RMSE of 20 runs; the second one is the RMSE of the average of the N_r runs,

$$RMSE_{AS} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (\overline{sim}_i - obs_i)^2} \tag{S11}$$

where $\overline{sim}_i$ is the mean simulated value at the time step i,

$$\overline{sim}_i = \frac{1}{N_r}\sum_{j=1}^{N_r} sim_{i,j} \tag{S12}$$

4.3 Differences between individual runs

For each case study, an individual run of the ML model provides a simulation that is not identical to the other runs even if the hyperparameters are the same, because the number of the parameters is so large that their calibration in the training phase produces results that depend on the initial (random) guess. Therefore, we introduced a measure (equation S7) for the robustness of the single prediction.

It is interesting to analyse a case where two runs have a very similar RMSE, but they differ significantly from each other. In Fig. S2, the simulated values in two runs (number 1 and 17) are shown: the mean difference between the two simulations is 0.308°C, although the RMSE is almost identical (1.221°C and 1.238°C, respectively).

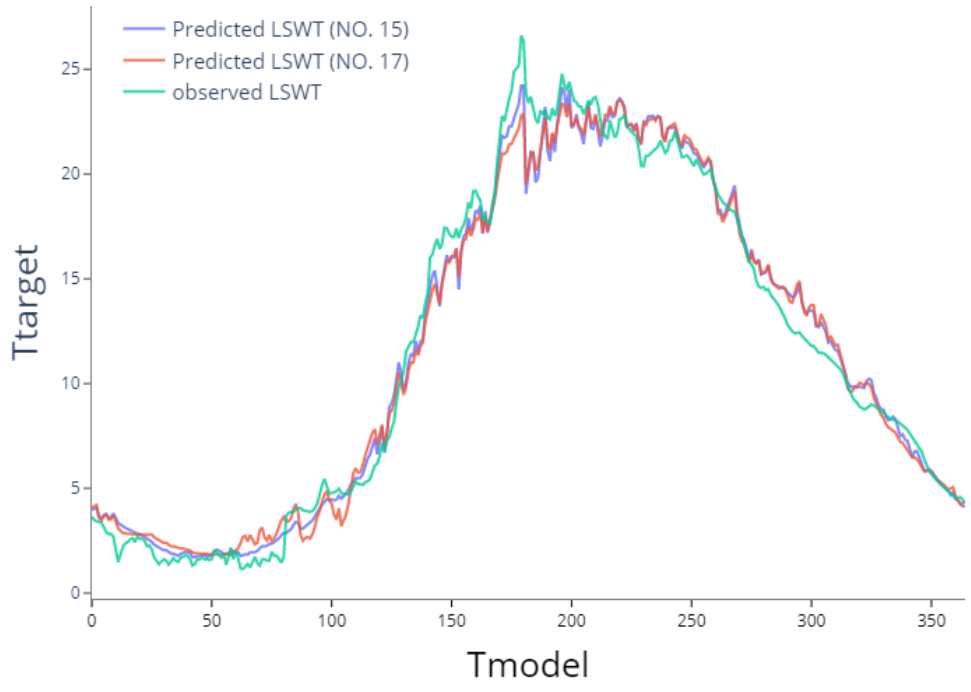

Figure S2. Simulated LSWT from two individual runs compared with observations. The two simulations have similar RMSE, although they differ from each other.

4.4 Analysis of the performances

Tables S3, S4, S5 and S6 report the list of all the metrics defined above to measure the performance of ML models, considering different inputs, pre-processing strategies, and ML algorithms.

Table S3. Metrics of the BPNN model for the shallow (depth = 5 m) lake, depending on the predictors used and separating the performances of the training and test data sets.

Depth = 5 m	$RMSE_A$ (°C)		MSE (°C^2)		MAE (°C)		NSE (-)		IA (-)		ME (°C)		σ_R (°C)		$RMSE_B$ (°C)		$RMSE_{AS}$ (°C)	
Variable(s)	train	test	train	test	train	test	train	test	train	test	train	test	train	test	train	test	train	test
AT	3.044	3.105	9.266	9.644	2.127	2.178	0.864	0.869	>0.999	>0.999	0.000	0.239	0.137	0.183	3.007	3.059	3.021	3.076
DOY	1.856	2.125	3.482	4.539	1.326	1.524	0.949	0.938	>0.999	>0.999	0.000	-0.738	0.257	0.257	1.782	2.065	1.796	2.069
SCDOY	1.781	2.070	3.172	4.286	1.262	1.470	0.953	0.942	>0.999	>0.999	0.000	-0.738	0.013	0.013	1.778	2.055	1.777	2.067
AT+SCDOY	1.312	1.460	1.720	2.131	0.921	1.013	0.975	0.971	>0.999	>0.999	0.000	-0.354	0.028	0.029	1.299	1.432	1.301	1.450
SWR	5.575	5.556	31.082	30.865	4.376	4.355	0.543	0.581	>0.999	>0.999	0.000	-0.311	0.017	0.021	5.572	5.548	5.574	5.554
LWR	5.810	5.904	33.757	34.857	4.628	4.771	0.504	0.527	>0.999	>0.999	0.000	0.128	0.079	0.099	5.792	5.885	5.803	5.896
RH	7.928	8.141	62.847	66.272	7.119	7.238	0.076	0.101	>0.999	>0.999	0.000	-0.693	0.040	0.034	7.922	8.133	7.925	8.139
WS	8.030	8.452	64.479	71.432	7.228	7.560	0.052	0.031	>0.999	>0.999	0.000	-0.765	0.070	0.071	8.011	8.431	8.025	8.448
R	8.227	8.594	67.677	73.849	7.507	7.754	0.005	-0.002	>0.999	>0.999	0.003	-0.716	0.002	0.003	8.224	8.590	8.226	8.593
AP	7.520	7.884	56.579	62.182	6.533	6.822	0.168	0.156	>0.999	>0.999	0.001	-0.492	0.642	0.618	7.472	7.840	7.479	7.846
AT + SCDOY + WS	1.313	1.452	1.723	2.108	0.921	1.006	0.975	0.971	>0.999	>0.999	0.000	-0.351	0.035	0.036	1.300	1.439	1.299	1.439
AT + SCDOY + AP	1.308	1.455	1.712	2.117	0.919	1.010	0.975	0.971	>0.999	>0.999	0.000	-0.344	0.040	0.042	1.298	1.432	1.293	1.440
AT + SCDOY + SWR	1.381	1.534	1.906	2.354	0.963	1.063	0.972	0.968	>0.999	>0.999	0.000	-0.400	0.034	0.035	1.369	1.514	1.368	1.523
AT + SCDOY + LWR	1.287	1.422	1.657	2.021	0.901	0.987	0.976	0.973	>0.999	>0.999	0.000	-0.360	0.041	0.043	1.270	1.399	1.271	1.406
AT + SCDOY + SWR + LWR	1.375	1.525	1.892	2.327	0.964	1.060	0.972	0.968	>0.999	>0.999	0.000	-0.370	0.048	0.049	1.357	1.501	1.358	1.509
AT + SCDOY + SWR + LWR + WS	1.369	1.509	1.874	2.277	0.956	1.047	0.972	0.969	>0.999	>0.999	0.000	-0.361	0.046	0.049	1.352	1.481	1.352	1.493
AT + SCDOY + SWR + LWR + WS + RH	1.371	1.497	1.879	2.240	0.960	1.038	0.972	0.970	>0.999	>0.999	0.000	-0.326	0.050	0.055	1.351	1.475	1.353	1.478
AT + SCDOY + SWR + LWR + WS + RH + R	1.358	1.484	1.844	2.201	0.951	1.028	0.973	0.970	>0.999	>0.999	0.000	-0.302	0.054	0.059	1.339	1.461	1.338	1.464
AT + SCDOY + SWR + LWR + WS + RH + R +AP	1.363	1.496	1.858	2.240	0.956	1.039	0.973	0.970	>0.999	>0.999	0.000	-0.285	0.065	0.071	1.340	1.461	1.339	1.473

Table S4. Metrics of the BPNN model for the deep (depth = 60 m) lake, depending on the predictors used and separating the performances of the training and test data sets.

Depth = 60 m	$RMSE_A$ (°C)		MSE (°C^2)		MAE (°C)		NSE (-)		IA (-)		ME (°C)		σ_R (°C)		$RMSE_B$ (°C)		$RMSE_{AS}$ (°C)	
Variable(s)	train	test	train	test	train	test	train	test	train	test	train	test	train	test	train	test	train	test
AT	3.629	3.911	13.187	15.303	2.837	2.989	0.779	0.757	>0.999	>0.999	0.000	0.154	0.258	0.266	3.587	3.881	3.596	3.878
DOY	1.596	1.746	2.654	3.185	1.186	1.353	0.956	0.950	>0.999	>0.999	-0.001	-0.743	0.482	0.482	1.436	1.579	1.474	1.644
SCDOY	1.434	1.579	2.056	2.492	1.041	1.223	0.966	0.960	>0.999	>0.999	0.000	-0.742	0.012	0.012	1.431	1.571	1.430	1.575
AT+SCDOY	1.160	1.292	1.345	1.670	0.868	1.007	0.978	0.974	>0.999	>0.999	0.000	-0.477	0.027	0.028	1.147	1.271	1.148	1.281
SWR	6.182	6.383	38.222	40.740	5.259	5.410	0.361	0.354	>0.999	>0.999	0.000	-0.401	0.005	0.006	6.181	6.380	6.182	6.382
LWR	5.300	5.285	28.086	27.932	4.236	4.243	0.530	0.557	>0.999	>0.999	0.000	0.081	0.068	0.084	5.289	5.273	5.293	5.277
RH	7.604	7.884	57.815	62.154	6.828	6.973	0.033	0.015	>0.999	>0.999	0.000	-0.802	0.081	0.096	7.593	7.782	7.598	7.878
WS	7.520	7.799	56.544	60.818	6.709	6.883	0.054	0.036	>0.999	>0.999	0.000	-0.767	0.030	0.030	7.515	7.791	7.518	7.797
R	7.703	7.946	59.333	63.142	6.958	7.060	0.008	-0.001	>0.999	>0.999	0.015	-0.705	0.037	0.042	7.699	7.940	7.700	7.944
AP	7.112	7.385	50.582	54.544	6.200	6.435	0.154	0.135	>0.999	>0.999	0.000	-0.515	0.056	0.050	7.102	7.373	7.108	7.382
AT + SCDOY + WS	1.151	1.294	1.326	1.675	0.862	1.007	0.978	0.973	>0.999	>0.999	0.000	-0.478	0.037	0.041	1.137	1.278	1.135	1.278
AT + SCDOY + AP	1.164	1.284	1.354	1.650	0.871	1.004	0.977	0.974	>0.999	>0.999	-0.001	-0.469	0.040	0.041	1.152	1.267	1.146	1.268
AT + SCDOY + SWR	1.204	1.328	1.449	1.764	0.895	1.034	0.976	0.972	>0.999	>0.999	0.000	-0.511	0.037	0.039	1.188	1.311	1.188	1.313
AT + SCDOY + LWR	1.153	1.282	1.329	1.644	0.859	1.003	0.978	0.974	>0.999	>0.999	0.000	-0.487	0.037	0.039	1.141	1.266	1.137	1.267
AT + SCDOY + SWR + LWR	1.189	1.322	1.413	1.747	0.888	1.029	0.976	0.972	>0.999	>0.999	0.000	-0.485	0.039	0.040	1.172	1.304	1.172	1.307
AT + SCDOY + SWR + LWR + WS	1.186	1.328	1.406	1.764	0.886	1.031	0.977	0.972	>0.999	>0.999	0.000	-0.486	0.049	0.050	1.165	1.304	1.165	1.309
AT + SCDOY + SWR + LWR + WS + RH	1.185	1.309	1.405	1.714	0.887	1.013	0.977	0.973	>0.999	>0.999	0.000	-0.432	0.054	0.059	1.165	1.290	1.162	1.286
AT + SCDOY + SWR + LWR + WS + RH + R	1.175	1.307	1.382	1.708	0.878	1.012	0.977	0.973	>0.999	>0.999	0.000	-0.419	0.060	0.070	1.151	1.262	1.149	1.280
AT + SCDOY + SWR + LWR + WS + RH + R +AP	1.176	1.301	1.382	1.692	0.880	1.008	0.977	0.973	>0.999	>0.999	0.001	-0.411	0.055	0.059	1.152	1.290	1.152	1.278

Table S5. Comparison of the metrics obtained with different pre-processing methods and considering the input of AT from previous days, for the shallow and deep lake (ML method: BPNN).

Depth = 5 m																		
Method	$RMSE_A$ (°C)		MSE (°C^2)		MAE (°C)		NSE (-)		IA (-)		ME (°C)		σ_R (°C)		$RMSE_B$ (°C)		$RMSE_{AS}$ (°C)	
set	train	test	train	test	train	test	train	test	train	test	train	test	train	test	train	test	train	test
Min-Max	1.312	1.460	1.720	2.131	0.921	1.013	0.975	0.971	>0.999	>0.999	0.000	-0.354	0.028	0.029	1.299	1.432	1.301	1.450
Moving Average + Min-Max	1.061	1.136	1.126	1.292	0.743	0.802	0.983	0.982	>0.999	>0.999	0.000	-0.097	0.026	0.032	1.047	1.108	1.049	1.123
AT from previous days	0.959	0.943	0.920	0.890	0.652	0.668	0.986	0.988	>0.999	>0.999	0.001	0.030	0.056	0.059	0.937	0.898	0.930	0.912
DWT	1.754	1.968	3.077	3.871	1.260	1.418	0.955	0.948	>0.999	>0.999	-0.454	0.024	0.039	1.747	1.948	1.747	1.958	-0.454
CWT	1.801	2.085	3.245	4.346	1.282	1.488	0.952	0.941	>0.999	>0.999	-0.740	0.025	0.025	1.796	2.073	1.795	2.079	-0.740
Depth = 60 m																		
Min-Max	1.160	1.292	1.345	1.670	0.868	1.007	0.978	0.974	>0.999	>0.999	0.000	-0.477	0.027	0.028	1.147	1.271	1.148	1.281
Moving Average + Min-Max	0.983	1.189	0.967	1.414	0.741	0.876	0.984	0.978	>0.999	>0.999	0.000	-0.202	0.031	0.036	0.973	1.153	0.968	1.174
AT from previous days	0.887	1.136	0.788	1.291	0.664	0.811	0.987	0.980	>0.999	>0.999	0.000	-0.088	0.067	0.072	0.869	1.103	0.849	1.104
DWT	1.365	1.538	1.863	2.364	1.011	1.160	0.969	0.963	>0.999	>0.999	0.000	-0.333	0.024	0.041	1.356	1.514	1.356	1.524
CWT	1.421	1.558	2.020	2.429	1.034	1.212	0.966	0.962	>0.999	>0.999	0.000	-0.742	0.023	0.023	1.415	1.552	1.413	1.551

Table S6. Comparison of the metrics obtained using different ML algorithms using AT and SCDOY as input (pre-processing: MM).

Depth = 5 m																		
Method	$RMSE_A$ (°C)		MSE (°C^2)		MAE (°C)		NSE (-)		IA (-)		ME (°C)		σ_R (°C)		$RMSE_B$ (°C)		$RMSE_{AS}$ (°C)	
set	train	test	train	test	train	test	train	test	train	set	train	test	train	test	train	test	train	test
DT	1.230	1.626	1.514	2.642	0.859	1.090	0.978	0.964	>0.999	>0.999	0.000	-0.340	0.000	0.002	1.230	1.623	1.230	1.625
RF	1.159	1.540	1.344	2.373	0.769	1.025	0.980	0.968	>0.999	>0.999	-0.033	-0.383	0.010	0.013	1.157	1.528	1.155	1.536
ERT	1.295	1.479	1.677	2.188	0.883	1.006	0.975	0.970	>0.999	>0.999	-0.033	-0.424	0.011	0.013	1.288	1.460	1.291	1.475
KNN	1.243	1.525	1.544	2.327	0.860	1.047	0.977	0.968	>0.999	>0.999	-0.006	-0.413	0.000	0.000	1.243	1.525	1.243	1.525
SVR	1.369	1.497	1.873	2.240	1.052	1.121	0.973	0.970	>0.999	>0.999	0.107	-0.259	0.000	0.000	1.369	1.497	1.369	1.497
MLPNN	1.319	1.467	1.739	2.153	0.926	1.022	0.974	0.971	>0.999	>0.999	-0.025	-0.383	0.043	0.042	1.299	1.446	1.302	1.453
LSTM	1.353	1.490	1.831	2.222	0.983	1.074	0.973	0.970	>0.999	>0.999	-0.024	-0.349	0.085	0.092	1.322	1.424	1.321	1.460
BPNN	1.312	1.460	1.720	2.131	0.921	1.013	0.975	0.971	>0.999	>0.999	0.000	-0.354	0.028	0.029	1.299	1.432	1.301	1.450
Depth = 60 m																		
DT	1.428	1.668	2.056	2.811	1.044	1.241	0.966	0.955	>0.999	>0.999	0.000	-0.472	0.750	0.916	1.264	1.447	1.143	1.377
RF	1.117	1.362	1.247	1.856	0.847	1.056	0.979	0.971	>0.999	>0.999	-0.001	-0.472	0.007	0.009	1.114	1.351	1.114	1.359
ERT	1.141	1.292	1.302	1.668	0.853	1.011	0.978	0.974	>0.999	>0.999	0.000	-0.522	0.003	0.004	1.136	1.285	1.140	1.290
KNN	1.093	1.311	1.196	1.719	0.808	1.026	0.980	0.973	>0.999	>0.999	-0.009	-0.510	0.000	0.000	1.093	1.311	1.093	1.311
SVR	1.171	1.302	1.371	1.695	0.914	1.025	0.977	0.973	>0.999	>0.999	0.107	-0.355	0.000	0.000	1.171	1.302	1.171	1.302
MLPNN	1.212	1.378	1.469	1.901	0.918	1.085	0.975	0.970	>0.999	>0.999	-0.083	-0.603	0.033	0.034	1.177	1.296	1.198	1.366
LSTM	1.286	1.264	1.656	1.598	0.986	0.962	0.972	0.975	>0.999	>0.999	0.405	-0.054	0.019	0.023	1.233	1.223	1.279	1.255
BPNN	1.160	1.292	1.345	1.670	0.868	1.007	0.978	0.974	>0.999	>0.999	0.000	-0.477	0.027	0.028	1.147	1.271	1.148	1.281

5.Hyperparameters optimized for the examined ML approaches

Eight of the ML methods considered in this paper were implemented in the Python environment using the libraries 'Keras' for Multilayer Perceptron Neural Network (MLPNN) and Long Short-Term Memory (LSTM) and 'Sklearn' for the other ML methods. Only for one method (ANFIS) we referred to a MATLAB library.

The hyperparameters were chosen by GA optimization. A more detailed explanation is available for 'Keras' in Chollet (2015) and for 'Sklearn' in Cournapeau (2007), where we also took most of the following description. In this document, we restrict the analysis to the hyperparameters that were calibrated and provide a brief description of the meaning and a table with adopted values. For the other hyperparameters that are not mentioned here, we used the default values suggested in the libraries.

Since GA choice of the best hyperparameters is approximate, the adopted hyperparameters may not be the optimal values in absolute terms. Different hyperparameters can be obtained by varying the population size and number of iterations in GA. The objective function is the absolute value of the difference of RMSE (°C) between the training and test data sets, plus the training RMSE (°C). With respect to this objective function, not only the chosen hyperparameters make the model to have lower RMSE but also the difference between the results of training and test data set would be the minimum.

- ### Decision Tree (DT)

The hyperparameters for DT (optimized in Table S7) are:

- **Criterion= ['MSE', 'friedman_MSE', 'MAE', 'Poisson']**

A measure of the quality of a split into two branches:

- 'MSE': mean squared error
- 'friedman_MSE': mean squared error with Friedman's improvement score for potential splits,
- 'MAE': mean absolute error
- 'poisson': reduction in Poisson deviance to find splits.

- **Max_depth**

The maximum depth of the tree. The 'none' is imposed, nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split (the minimum required number of samples for splitting an internal node) samples, where min_samples_split is a hyperparameter to be defined.

- **Max_features= [int, float, 'auto', 'sqrt', 'log2']**

The number of features to consider each time to make the split decision.

- int: max_features at each split

- float: max_features is a fraction at each split; max_features * n_features (the number of features seen during fit) is the value to consider as features.
- 'auto': max_features is n_features.
- 'SQRT': max_features is the square root of n_features.
- 'log2': max_features is logarithm of n_features.
- None: max_features is n_features.

Table S7. Optimized values of DT hyperparameters.

Lake's depth (m)	Criterion	max_depth	max_features
5	'MSE'	8	'auto'
60	'friedman_MSE'	7	'log2'

Random Forest (RF)

The hyperparameters for RF (optimized in Table S8) are:

- **N_estimators**

The number of trees in the forest.

- **Criterion= ['MSE', 'MAE']**

The function to measure the quality of a split, where 'MSE' and 'MAE' are explained in previous section.

- **Max_depth**

The maximum depth of the tree. 'None' means that the nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

Table S8. Optimized values of RF hyperparameters.

Lake's depth (m)	n_estimators	Criterion	max_depth
5	42	'MAE'	9
60	26	'MSE'	7

Extremely Randomized Tree (ERT)

The explanation of the hyperparameters of ERT is similar to RF. The optimized values are reported in Table S9.

Table S9. Optimized values of ERT hyperparameters.

Lake's depth (m)	n_estimators	Criterion	max_depth
5	85	'MAE'	7
60	181	'MSE'	7

K-nearest neighbour (KNN)

The hyperparameters for KNN (optimized in Table S10) are:

- **N_neighbors**

Number of neighbours required for each sample.

- **Weights= [Uniform', 'Distance', Callable]**

Weight function used in prediction. Possible values:

- 'Uniform': all points in each neighbourhood are weighed equally.
- 'Distance': weight points by the inverse of their distance. In this case, closer neighbours of a query point will have a greater influence than neighbours which are further away.
- Callable: a user-defined function which accepts an array of distances, and returns an array of the same shape containing the weights.

- **Algorithm= ['auto', 'Ball_Tree', 'KD_Tree', 'Brute']**

Algorithm used to compute the nearest neighbours:

- 'Ball_tree': recursively divides the data into nodes defined by a centroid C and radius r, such that each point in the node lies within the hyper-sphere defined by r and C.
- 'KD_Tree': a binary tree structure which recursively partitions the parameter space along the data axes, dividing it into nested orthotropic regions into which data points are filed.
- 'Brute': the brute-force computation of distances between all pairs of points in the dataset.
- 'auto': decide the most appropriate algorithm based on the values passed to fit method.

- **Leaf_size**

Leaf size passed to BallTree or KDTree. This can affect the speed of the construction and query, as well as the memory required to store the tree. The optimal value depends on the nature of the problem.

- **P**

Power parameter for the Minkowski metric.

- P = 1: Manhattan_distance,
- P = 2: Euclidean_distance,
- Arbitrary P: Minkowski_distance.

Table S10. Optimized values of KNN hyperparameters.

Lake's depth (m)	N_neighbors	Weights	Algorithm	Leaf_size	P
5	23	'uniform'	'KD_Tree'	22	3
60	20	'uniform'	'KD_Tree'	22	3

Support Vector Regression (SVR)

The hyperparameters for SVR (optimized in Table S11) are:

- **Kernel= ['Linear', 'Poly', 'RBF', 'Sigmoid', 'Precomputed']**

The choice of the kernel type to be used in the algorithm. The first two kernels are linear and polynomial, and the other kernels are:

- 'RBF': Radial Basis Function (RBF) kernel.
- 'Sigmoid': based on the hyperbolic tangent,

$$K(X, Y) = \tanh(\alpha X^T Y + c) \quad \text{(S13)}$$

where α is slope, and C is the constant term; X and Y are inputs of the model and observed outputs, respectively.

- 'Precomputed': used to pre-compute the kernel matrix from data matrices; matrix should be a square array of size number of samples.

- **Degree**

Degree of the polynomial kernel function ('Poly'). Ignored by all other kernels.

- **Gamma= ['Scale', 'auto']**

Kernel coefficient for 'RBF', 'Poly' and 'Sigmoid':

$$'\text{scale}' = \frac{1}{\text{n\_features} \times \text{X.var}()} \quad \text{(S14)}$$

Where '.var' is the variance, or

$$'\text{auto}' = \frac{1}{\text{n\_features}} \quad \text{(S15)}$$

- **C**

Regularization parameter. The strength of the regularization is inversely proportional to C. Must be strictly positive.

- **Epsilon**

Epsilon in the epsilon-SVR model. It specifies the epsilon-tube within which no penalty is associated in the training loss function with points predicted within a distance epsilon from the actual value.

- **Cache_size**

Specify the size of the kernel cache.

Table S11. Optimized values of SVR hyperparameters.

Lake's depth (m)	Kernel	Degree	Gamma	C	Epsilon	cache_size
5	'Poly'	5	'auto'	93.16	0.0555	162.9
60	'RBF'	1	'auto'	158.4	0.0555	161.2

• Multilayer Perceptron Neural Network (MLPNN)

The meaning of the number of layers and neurons is explained in Appendix B of the main text (see the section for ANN). The hyperparameters for MLPNN (optimized in Table S12) are:

- **Number of hidden layers**

The number of neurons in the hidden layer which is also used for LSTM and BPNN in the following methods.

- **Activation function= ['RELU'', 'Sigmoid', 'Softmax', 'Softplus', 'Softsign', 'Tanh', 'SELU', 'ELU', 'Exponential']**
  - 'RELU'': linear unit activation function
  - 'Softmax': converts a real vector to a vector of categorical probabilities.
  - 'Softplus':

$$\text{softplus}(x) = \log(\exp(x) + 1) \quad \text{(S16)}$$

- ‘Softsign’:

$$\text{softsign}(x) = x / (\text{abs}(x) + 1) \quad \text{(S17)}$$

- ‘SELU’: Scaled Exponential Linear Unit
- ’ELU’: Exponential Linear Unit.

- **Optimization= [‘SGD’, ‘RMSprop’, ‘Adam’, ‘Adadelta’, ‘Adagrad’, ‘Adamax’, ‘Nadam’, ‘Ftrl’]**
  - ‘SGD’: Stochastic gradient descent (with momentum) optimizer.
  - ‘RMSprop’: maintains a moving (discounted) average of the square of gradients and divides the gradient by the root of this average.
  - ‘Adam’: a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments, proposed by Kingma and Ba (2014).
  - ‘Adadelta’: a stochastic gradient descent method that is based on adaptive learning rate per dimension to address the continual decay of learning rates throughout training and the need for a manually selected global learning rate.
  - ‘Adagrad’: with parameter-specific learning rates, which are adapted relative to how frequently a parameter gets updated during training. The more updates a parameter receives, the smaller the updates.
  - ‘Adamax’: a variant of Adam based on the infinity norm. Adamax is sometimes superior to adam, especially in models with embeddings.
  - ‘Nadam’: much like Adam is essentially RMSprop with momentum, Nadam is Adam with Nesterov momentum.
  - ‘Ftrl’: obtained from McMahan et al. (2013)

- **Learning rate**

The parameter that controls how much to change the model in response to the estimated error each time the model weights are updated.

- **Batch size**

The number of samples to work through before updating the internal model parameters.

- **Number of epochs**

The number of times that the whole set of patterns is presented to the network affected by other hyperparameters such as number of training data, number of hidden layers and number of neurones (Rafiq et al., 2011).

Table S12. Optimized values of MLPNN hyperparameters.

Lake's depth (m)	Number of hidden layers	Number of neurons	Activation function	Dropout	Optimization	Learning rate	Batch size	Number of epochs
5	1	9	'RELU''	0	SGD	0.08465	43	188
60	15	14	'RELU''	0.007955	SGD	0.10555	43	161

- Long Short-Term Memory (LSTM)

The hyperparameters are similar to MLPNN section. The optimized values are reported in Table S13.

Table S13. Optimized values of LSTM hyperparameters.

Lake's depth (m)	Number of hidden layers	Number of neurons	Activation function	Dropout	Optimization	Learning rate	Batch size	Number of epochs
5	1	8	'Tanh'	0.02787	Adam	0.06281	177	149
60	1	8	'Tanh'	0	Adam	0.06281	177	149

- Back Propagation Neural Network (BPNN)

The hyperparameters for BPNN (optimized in Table S14) are:

- **Activation= ['Identity', 'Logistic', 'Tanh', 'RELU'']**

Activation function for the hidden layer(s):

- 'Identity': useful to implement linear bottleneck, returns the following equation:

$$f(x) = x \tag{S18}$$

- 'Logistic': the logistic sigmoid function, returns the following equation:

$$f(x) = 1 / (1 + \exp(-x)) \tag{S19}$$

- 'Tanh': the hyperbolic tan function, returns the following equation:

$$f(x) = \tanh(x) \tag{S20}$$

- 'RELU'': the rectified linear unit function, returns the following equation:

$$f(x) = \max(0, x) \tag{S21}$$

- **Solver (Optimizer)**= ['LBFGS', 'SGD', 'Adam']

The solver for weight optimization:

- 'LBFGS': an optimizer in the family of quasi-Newton methods.
- 'SGD': stochastic gradient descent.
- 'Adam': explained in MLPNN.

- **Batch_size**

Size of minibatches, which the gradient is calculated across the entire batch before updating weights, for stochastic optimizers. If the solver is 'LBFGS', the classifier will not use minibatch. When set to "auto", batch_size is equal to minimum of 200 and n_samples.

- **learning_rate= ['Constant', 'Invscaling', 'Adaptive']**

Learning rate schedule for weight updates:

- 'Constant': a constant learning rate given by 'learning_rate_init'.
- 'Invscaling': gradually decreases the learning rate at each time step 't' using an inverse scaling exponent of 'power_t'.
- 'Adaptive': keeps the learning rate constant to 'learning_rate_init' as long as training loss keeps decreasing. Each time two consecutive epochs fail to decrease training loss or fail to increase validation score, the current learning rate is divided by 5.

- **Learning_rate_init**

The initial learning rate used. It controls the step-size in updating the weights.

- **Max_iter**

Maximum number of iterations. The solver iterates until convergence (determined by 'tol') or this number of iterations. For stochastic solvers ('SGD, 'Adam'), note that this determines the number of epochs (how many times each data point will be used), not the number of gradient steps.

Table S14. Optimized values of BP hyperparameters

Lake's depth (m)	Number of hidden layers	Number of neurons	Activation function	Optimization	Learning rate	Learning rate initial	Maximum iteration
5	19	20	'RELU''	'LBFGS'	'Adaptive'	0.00431	211
60	15	19	'RELU''	'LBFGS'	'Adaptive'	0.00471	196

• Adaptive Network-based Fuzzy Inference System (ANFIS)

The features considered in this method are taken from the MATLAB toolbox for ANFIS. Based on our available data, the best hyperparameters are shown in Table S15.

- **Input Membership function**

The function based on the fuzzy logic principles train the model set to calculate the parameters of membership function, which represents the degree of truth in fuzzy logic (Opeyemi and Justice, 2012). The following functions are introduced:

- Gaussian (Gaussmf):

$$f(x) = e^{\frac{-(x-c)^2}{2\sigma^2}} \quad \text{(S22)}$$

where σ is the standard deviation, c is the mean.

- Generalized bell (Gbellmf):

$$f(x) = \frac{1}{1 + \left|\frac{x - c}{a}\right|^{2b}} \quad \text{(S23)}$$

where a, b, and c are parameters.

- Sigmoid (Sigmf):

$$f(x) = \frac{1}{1 + e^{-a_k(x - c_k)}} \quad \text{(S24)}$$

where a_k and c_k parameters.

- Triangular membership function (Trimf):

$$f(x) = \begin{cases} 0, & x \le a \\ \frac{x - a}{b - a}, & a \le x \le b \\ \frac{c - x}{c - b}, & b \le x \le c \\ 0, & c \le x \end{cases} \quad \text{(S25)}$$

where a, b, and c are parameters.

- **Output Membership function**

The output type of the membership function should be chosen in one of these two ways:

- Linear
- Constant

- **Optimisation method**

Method used to train the model. Two optimization methods are used to reduce the error:

- Backpropagation: for all parameters (as explained in Appendix B).
- Hybrid method: consisting of backpropagation for the parameters associated with the input membership functions, and least squares estimation for the parameters associated with the output membership functions.

- **Epochs**

A measure of the number of times that the model is trained to update the weights.

Table S15. Optimized values of BP hyperparameters

Lake's depth (m)	Input MF	Output MF	Optimization	Epochs
5	Trimf	Linear	Hybrid	100
60	Trimf	Linear	hybrid	100

References

Caissie, D., El-Jabi, N. and Satish, M.G., 2001. Modelling of maximum daily water temperatures in a small stream using air temperatures. Journal of hydrology, 251(1-2), pp.14-28. https://doi.org/10.1016/S0022-1694(01)00427-9

Chenard, J.F. and Caissie, D., 2008. Stream temperature modelling using artificial neural networks: application on Catamaran Brook, New Brunswick, Canada. Hydrological Processes: An International Journal, 22(17), 3361-3372. https://doi.org/10.1002/hyp.6928

Chollet, F., 2015, Keras, https://keras.io/

Cournapeau, D., 2007, scikit-learn, https://scikit-learn.org/

DeWeber, J.T. and Wagner, T., 2014. A regional neural network ensemble for predicting mean daily river water temperature. Journal of Hydrology, 517, 187-200. https://doi.org/10.1016/j.jhydrol.2014.05.035

Erickson, T.R. and Stefan, H.G., 2000. Linear air/water temperature correlations for streams during open water periods. Journal of Hydrologic Engineering, 5(3), 317-321. https://doi.org/10.1061/(ASCE)10840699(2000)5:3(317)

Grbić, R., Kurtagić, D. and Slišković, D., 2013. Stream water temperature prediction based on Gaussian process regression. Expert systems with applications, 40(18), 7407-7414. doi:10.1016/j.eswa.2013.06.077

Goyal, M.K., Burn, D.H. and Ojha, C.S.P., 2012. Evaluation of machine learning tools as a statistical downscaling tool: temperatures projections for multi-stations for Thames River Basin, Canada. Theoretical and Applied Climatology, 108(3-4), 519-534. https://doi.org/10.1007/s00704-011-0546-1

Graf, R., Zhu, S. and Sivakumar, B., 2019. Forecasting river water temperature time series using a wavelet–neural network hybrid modelling approach. Journal of Hydrology, 578, 124115. https://doi.org/10.1016/j.jhydrol.2019.124115

Hadzima-Nyarko, M., Rabi, A. and Šperac, M., 2014. Implementation of artificial neural networks in modeling the water-air temperature relationship of the River Drava. Water resources management, 28(5), 1379-1394. doi: 10.1007/s11269-014-0557-7

Holthuijzen, M.F., 2017. A comparison of five statistical methods for predicting stream temperature across stream networks. Utah State University.

Kingma, D.P. and Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Laanaya, F., St-Hilaire, A. and Gloaguen, E., 2017. Water temperature modelling: comparison between the generalized additive model, logistic, residuals regression and linear regression models. Hydrological sciences journal, 62(7), 1078-1093. doi: 10.1080/02626667.2016.1246799

Li, H., Deng, X., Kim, D.Y. and Smith, E.P., 2014. Modeling maximum daily temperature using a varying coefficient regression model. Water Resources Research, 50(4), 3073-3087. https://doi.org/10.1002/2013WR014243

Lu, H. and Ma, X., 2020. Hybrid decision tree-based machine learning models for short-term water quality prediction. Chemosphere, 249, p.126169. https://doi.org/10.1016/j.chemosphere.2020.126169

McMahan, H.B., Holt, G., Sculley, D., Young, M., Ebner, D., Grady, J., Nie, L., Phillips, T., Davydov, E., Golovin, D. and Chikkerur, S., 2013, August. Ad click prediction: a view from the trenches. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1222-1230). https://doi.org/10.1145/2487575.2488200

Mohseni, O. and Stefan, H.G., 1999. Stream temperature/air temperature relationship: a physical interpretation. Journal of hydrology, 218(3-4), 128-141. https://doi.org/10.1016/S0022-1694(99)00034-7

Mohseni, O., Stefan, H.G. and Erickson, T.R., 1998. A non-linear regression model for weekly stream temperatures. Water Resources Research, 34(10), 2685-2692. https://doi.org/10.1029/98wr01877

Napiorkowski, M.J., Piotrowski, A.P. and Napiorkowski, J.J., 2014. Stream temperature forecasting by means of ensemble of neural networks: Importance of input variables and ensemble size. In River Flow, 2017-2025. https://doi.org/10.1201/b17133-269

Opeyemi, O. and Justice, E.O., 2012. Development of neuro-fuzzy system for early prediction of heart attack. Information Technology and Computer Science, 9(9), 22-28. https://doi.org/10.5815/ijitcs.2012.09.03

Piotrowski, A.P., Napiorkowski, J.J. and Piotrowska, A.E., 2020. Impact of deep learning-based dropout on shallow neural networks applied to stream temperature modelling. Earth-Science Reviews, 201, 103076. https://doi.org/10.1016/j.earscirev.2019.103076

Piotrowski, A.P., Napiorkowski, M.J., Kalinowska, M., Napiorkowski, J.J. and Osuch, M., 2016. Are evolutionary algorithms effective in calibrating different artificial neural network types for streamwater temperature prediction? Water resources management, 30(3), 1217-1237. https://doi.org/10.1007/s11269-015-1222-5

Piotrowski, A.P., Napiorkowski, M.J., Napiorkowski, J.J. and Osuch, M., 2015. Comparing various artificial neural network types for water temperature prediction in rivers. Journal of Hydrology, 529, 302-315. https://doi.org/10.1016/j.jhydrol.2015.07.044

Piotrowski, A.P., Osuch, M., Napiorkowski, M.J., Rowinski, P.M. and Napiorkowski, J.J., 2014. Comparing large number of metaheuristics for artificial neural networks training to predict water temperature in a natural river. Computers & Geosciences, 64, 136-151. https://doi.org/10.1016/j.cageo.2013.12.013

Qiu, R., Wang, Y., Wang, D., Qiu, W., Wu, J. and Tao, Y., 2020. Water temperature forecasting based on modified artificial neural network methods: Two cases of the Yangtze River. Science of The Total Environment, p.139729. https://doi.org/10.1016/j.scitotenv.2020.139729

Rabi, A., Hadzima-Nyarko, M. and Šperac, M., 2015. Modelling river temperature from air temperature: case of the River Drava (Croatia). Hydrological sciences journal, 60(9), 1490-1507. https://doi.org/10.1080/02626667.2014.914215

Radulescu, V., 2020. Application of the Neural Networks in Prediction of the Thermal Flow on the Jiu River. In Heat Transfer Summer Conference (Vol. 83709, p. V001T04A005). American Society of Mechanical Engineers. https://doi.org/10.1115/HT2020-9043

Rafiq, M.Y., Bugmann, G. and Easterbrook, D.J., 2001. Neural network design for engineering applications. Computers & Structures, 79(17), pp.1541-1552. https://doi.org/10.1016/j.compstruc.2005.03.019

Rehana, S., 2019. River water temperature modelling under climate change using support vector regression. In Hydrology in a Changing World, 171-183. Springer, Cham. https://doi.org/10.1007/978-3-030-02197-9_8

Rivers-Moore, N.A., Bezuidenhout, C.N. and Jewitt, G.P., 2005. Modelling highly variable daily maximum water temperatures in a perennial South African river system. African Journal of Aquatic Science, 30(1), 55-63. https://doi.org/10.2989/16085910509503835

Sahoo, G.B., Schladow, S.G. and Reuter, J.E., 2009. Forecasting stream water temperature using regression analysis, artificial neural network, and chaotic non-linear dynamic models. Journal of Hydrology, 378(3-4), 325-342. https://doi.org/10.1016/j.jhydrol.2009.09.037

St-Hilaire, A., Ouarda, T.B., Bargaoui, Z., Daigle, A. and Bilodeau, L., 2012. Daily river water temperature forecast model with a k-nearest neighbour approach. Hydrological Processes, 26(9), 1302-1310. doi:10.1002/hyp.8216

Temizyurek, M. and Dadaser-Celik, F., 2018. Modelling the effects of meteorological parameters on water temperature using artificial neural networks. Water Science and Technology, 77(6), 1724-1733. https://doi.org/10.2166/wst.2018.058

Trinh, N.X., Trinh, T.Q., Phan, T.P., Thanh, T.N. and Thanh, B.N., 2019. Water Temperature Prediction Models in Northern Coastal Area, Vietnam. Asian Review of Environmental and Earth Sciences 6 (1) 2313-8173. doi: 10.20448/journal.506.2019.61.1.8.

Wenxian, G., Hongxiang, W., Jianxin, X. and Wensheng, D., 2010, May. PSO-BP neural network model for predicting water temperature in the middle of the Yangtze river. In 2010 International Conference on Intelligent Computation Technology and Automation, 2, 951-954. IEEE. doi: 10.1109/ICICTA.2010.501

Willmott, C.J., 1981. On the validation of models. Physical geography, 2(2), pp.184-194. https://doi.org/10.1080/02723646.1981.10642213

Zhu, S. and Heddam, S., 2019. Modelling of Maximum Daily Water Temperature for Streams: Optimally Pruned Extreme Learning Machine (OPELM) versus Radial Basis Function Neural Networks (RBFNN). Environmental Processes, 6(3), 789-804. doi: 10.1007/s40710-019-00385-8

Zhu, S., Bonacci, O., Oskoruš, D., Hadzima-Nyarko, M. and Wu, S., 2019. Long term variations of river temperature and the influence of air temperature and river discharge: case study of Kupa River watershed in Croatia. Journal of Hydrology and Hydromechanics, 67(4), 305-313. https://doi.org/10.2478/johh-2019-0019

Zhu, S., Hadzima-Nyarko, M., Gao, A., Wang, F., Wu, J. and Wu, S., 2019. Two hybrid data-driven models for modeling water-air temperature relationship in rivers. Environmental Science and Pollution Research, 26(12), 12622-12630. https://doi.org/10.1007/s11356-019-04716-y

Zhu, S., Heddam, S., Nyarko, E.K., Hadzima-Nyarko, M., Piccolroaz, S., and Wu, S., 2019. Modeling daily water temperature for rivers: comparison between adaptive neurofuzzy inference systems and artificial neural networks models. Environmental Science and Pollution Research 26 (1), 402–420. doi: 10.1007/s11356-018-3650-2

Zhu, S., Heddam, S., Wu, S., Dai, J. and Jia, B., 2019. Extreme learning machine-based prediction of daily water temperature for rivers. Environmental Earth Sciences, 78(6), 202. doi: 10.1007/s12665-019-8202-7

Zhu, S., Nyarko, E.K. and Hadzima-Nyarko, M., 2018. Modelling daily water temperature from air temperature for the Missouri River. PeerJ, 6, e4894. doi: 10.7717/peerj.4894

Zhu, S., Nyarko, E.K., Hadzima-Nyarko, M., Heddam, S. and Wu, S., 2019. Assessing the performance of a suite of machine learning models for daily river water temperature prediction. PeerJ, 7, e7065. doi: 10.7717/peerj.7065

Zhu, S., Piotrowski, A.P., 2020. River/stream water temperature forecasting using artificial intelligence models: a systematic review. Acta Geophysica, 68(5), pp.1433–1442. doi:10.1007/s11600-020-00480-7. https://doi.org/10.1007/s11600-020-00480-7