

1 **From Precipitation to Prediction: Integrating**
2 **ConvLSTM Models for Comprehensive River Runoff**
3 **Forecasting**

4 **Florian Börgel¹, Sven Karsten¹, Karoline Rummel¹**

5 ¹Leibniz-Institute for Baltic Sea Research Warnemünde,

6 **Abstract**
 7 a

8 **Plain Language Summary**
 9 b

10 **1 Introduction**

11 River runoff is an important component of the global water cycle as it comprises
 12 about one third of the precipitation over land areas (Hagemann et al., 2020). In
 13 the context of climate change studies, river runoff is usually generated in two ways.
 14 First, river runoff as input for ocean models can be created using hydrological mod-
 15 els such as the Hydrological Discharge (HD) model (Hagemann et al., 2020). HD
 16 models calculate the water balance using hydrological processes (e.g. snow, glaciers,
 17 soil moisture, groundwater contribution). It represents a complex forecasting tool
 18 that uses underlying physical processes. A different approach would use data-based
 19 models that integrate statistical correction, using the land surface schemes of global
 20 or regional climate models.

21 The relatively recent rise of machine learning (ML) models has been mostly ex-
 22 plored for river runoff forecasting, as accurate runoff forecasting, especially over
 23 extended periods, is pivotal for effective water resources management (Fang et al.,
 24 2019; Tan et al., 2018; Yang et al., 2018). Common approaches employ artificial
 25 neural networks, support vector machines, adaptive neuro-fuzzy inference systems,
 26 and notably, Long Short-Term Memory (LSTM) neural networks that have gained
 27 traction for long-term hydrological forecasting due to their excellent performance
 28 (Humphrey et al 2016, Huang et al 2014, Ashrafi et al 2017, Yuan et al 2018, Xu et
 29 al 2021).

30 LSTM networks, an evolution of the classical Recurrent Neural Networks (RNNs),
 31 have shown stability and efficacy in sequence-to-sequence predictions, such as using
 32 climatic indices for rainfall estimation or long-term hydrological forecasting. How-
 33 ever, a limitation of LSTMs is their inability to effectively capture two-dimensional
 34 structures, an area where Convolutional Neural Networks (CNNs) excel. Recognizing
 35 this limitation ([shi2015?](#)) proposed a convolutional LSTM (ConvLSTM) archi-
 36 tectures, which combines the strengths of both LSTM and CNN. The ConvLSTM
 37 network has been proven useful for precipitation nowcasting ([shi2015?](#)) or for river
 38 runoff forecasting [[\(ha2021?\)](#); [@niu2020](#)].

39 In the following we will show that in absence of a fully functioning hydrogolocial
 40 model, that also uses a rather complex parametrization, ConvLSTM also represent
 41 a robust way to predict multiple rivers at once for any given period using only at-
 42 mospheric forcing. In this work we use the Baltic Sea catchment to illustrate our
 43 approach, while in principle the methodology we propose is universally applicable
 44 across various geographic regions. The Baltic Sea serves as a challenging example
 45 due to its unique hydrological characteristics, being nearly decoupled from the open
 46 ocean (see Figure). As a consequence, the salinity of the Baltic Sea is driven to
 47 a large part by freshwater supply from rivers. More generally, the freshwater in-
 48 put into the Baltic Sea comes either as river runoff or a positive net precipitation
 49 (precipitation minus evaporation) over the sea surface. The net precipitation ac-
 50 counts for 11 % and the river input for 89 % of the total freshwater input (Meier
 51 and Doescher, 2002). Modeling the Baltic Sea is therefore to a large part the result
 52 of the quality of the river input, that is used for the simulation. This makes the ac-
 53 curate modeling of river runoff especially critical for simulations pertaining to the
 54 Baltic Sea.

55 In this work we will, we present a ConvLSTM architecture that is able to predict
 56 daily river runoff for 97 rivers across the Baltic Sea catchment.

57 2 Methods

58 2.1 Runoff data used for training

59 The runoff data covering the period 1979 to 2011 is based on an E-HYPE hindcast
 60 simulation that was forced by a regional downscaling of ERA-Interim (**dee2011?**)
 61 with RCA3 (**theross?**) and implemented into NEMO-Nordic (**hordoir2019?**) as a
 62 mass flux. For the periods before (1961 to 1978) and after (2012 to 2018) additional
 63 spatial temporal corrections have been applied to the runoff data, and have therefore
 64 been ignored. For more information see (**gröger2022?**) and references therein.

65 2.2 Atmospheric Forcing

66 The UERRA-HARMONIE regional reanalysis dataset was developed as part of
 67 the FP7 UERRA project (Uncertainties in Ensembles of Regional Re-Analyses,
 68 <http://www.uerra.eu/>,). The UERRA-HARMONIE system represents a comprehen-
 69 sive, high-resolution reanalysis covering a wide range of essential climate variables.
 70 This dataset encompasses data on air temperature, pressure, humidity, wind speed
 71 and direction, cloud cover, precipitation, albedo, surface heat fluxes, and radiation
 72 fluxes from January 1961 to July 2019. With a horizontal resolution of 11 km and
 73 analyses conducted at 00 UTC, 06 UTC, 12 UTC, and 18 UTC, it also provides
 74 hourly resolution forecast model data. UERRA-HARMONIE is accessible through
 75 the Copernicus Climate Data Store (CDS, <https://cds.climate.copernicus.eu/#!/home>), initially produced during the UERRA project and later transitioned to
 76 the Copernicus Climate Change Service (C3S, <https://climate.copernicus.eu/copernicus-regionalreanalysis-europe>).
 77

78 2.3 Ocean Model

79 To simulate the Baltic Sea, we use a coupled 3-dimensional ocean model Baltic Sea,
 80 called the Modular Ocean Model (MOM). This model uses a finite-difference method
 81 to solve the full set of primitive equations to calculate the motion of water and the
 82 transport of heat and salt. The K-profile parameterization (KPP) was used as tur-
 83 bulence closure scheme. The model's western boundary opens into the Skagerrak
 84 and connects the Baltic Sea to the North Sea. The maximum depth was set at 264
 85 meters. A more detailed description of the setup can be found in (**gröger2022?**).
 86

87 2.4 LSTM network

88 Before turning to the convolutional LSTM, the simpler architecture of the plain
 89 LSTM is examined and should serve as a role model for the later consideration of
 90 the full ConvLSTM.

91 The Long Short-Term Memory (LSTM), a specialized form of Recurrent Neu-
 92 ral Networks (RNNs), is specifically tailored for modeling temporal sequences
 93 $\vec{X}^t[1], \dots \vec{X}^t[\tau], \dots \vec{X}^t[N_\tau]$ of N_τ input quantities $\vec{X}^t[\tau] = (X_k^t[\tau]) \in \mathbb{R}^{N_k}$. The se-
 94 quence is taken from a dataset given in form of a time series $\{\vec{X}^t\}$ and the endpoint
 95 should coincide with the specific element in the time series connected to time t ,
 96 i.e. $\vec{X}^t[N_\tau] \equiv \vec{X}^t$. The N_k is the number of used input “channels” and can, for exam-
 97 ple, represent different measurable quantities. Its unique design allows it to adeptly
 98 handle long-range dependencies, setting it apart from traditional RNNs in terms of
 99 accuracy (see Figure 1).

100 This performance in modeling long-range dependencies has been validated in
 101 various studies. The key component of LSTM's innovation is its memory cell,
 102 $\vec{C}^t[\tau] = (C_h^t[\tau]) \in \mathbb{R}^{N_h}$, which stores state information, also referred to as long-
 103 term memory. The index $\tau \in [1, N_\tau]$ corresponds to the τ -th element in the input
 104 sequence. The cell state is a vector consists of elements, each associated with one
 105 of the N_h hidden layers, labeled by h . This vector is determined through several

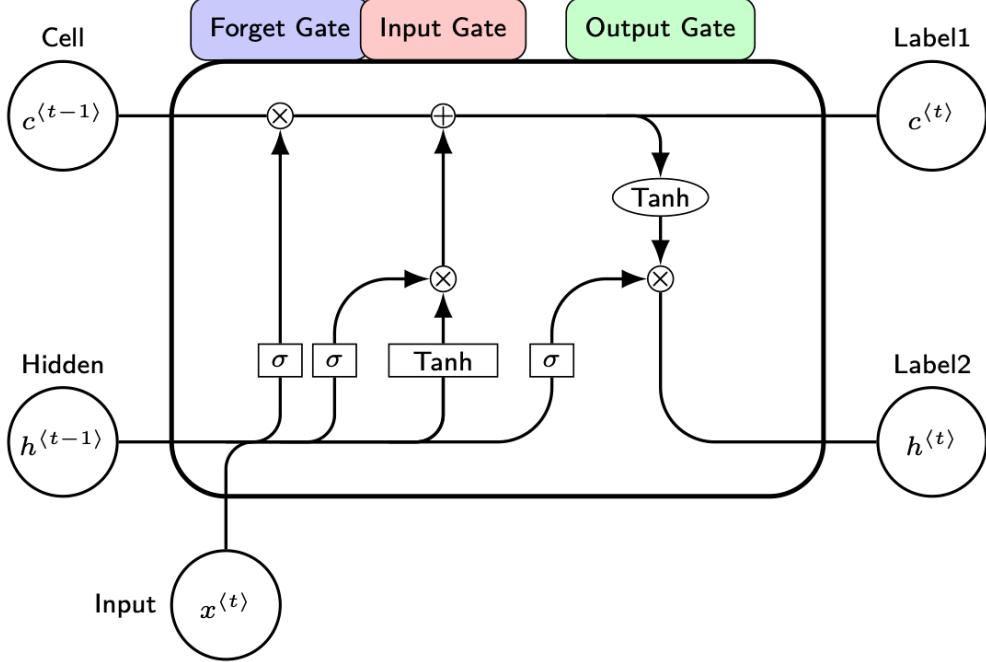


Figure 1: Inner structure of a Long Short-Term Memory Cell

self-parameterized gates, all in the same vector space as $\vec{C}^t[\tau]$. The forget gate $\vec{F}^t[\tau]$ defines the percentage of the previous long-term memory status $\vec{C}^t[\tau-1]$ that should be retained stored. The input gate $\vec{I}^t[\tau]$ decides how much of the input is added to the long-term memory, forming the updated cell state. The so-called output gate $\vec{O}^t[\tau]$ then determines how much of the latest cell output, $\vec{C}^t[\tau]$, is propagated to the final state, $\vec{H}^t[\tau]$ representing the updated short-term memory of the hidden state.

For a fixed point τ in the sequence, the action of a LSTM cell, i.e. the connection between the input $\vec{X}^t[\tau]$, the various gates and the state vectors, is given as follows. First, the elements of the input sequence together with the hidden state are mapped onto auxiliary gate vectors, collectively denoted by $\vec{g}^t[\tau] = (g_h^t[\tau]) \in \mathbb{R}^{N_h}$, via

$$g_h^t[\tau] = \mathcal{M}_{hk}^g X_k^t[\tau] + \mathcal{N}_{hh'}^g H_{h'}^t[\tau-1] + \mathcal{B}_h^g ,$$

where $g = i, f, o, c$ stands for the input, forget, output and cell-state gate, respectively and Eistein's summation convention is employed, i.e. indices that appear twice are summed over. The calligraphic symbols \mathcal{M}_{hk}^g , $\mathcal{N}_{hh'}^g$ and \mathcal{B}_h^g are the free parameters of the network that are optimized for the given problem during the training, which is at the heart of the any machine learning approach. The matrix $\mathcal{M}^g = (\mathcal{M}_{hk}^g) \in \mathbb{R}^{N_h \times N_k}$ can be interpreted as a Markovian-like contribution of the current input $\vec{X}^t[\tau]$ to the gates, whereas the $\mathcal{N}^g = (\mathcal{N}_{hh'}^g) \in \mathbb{R}^{N_h \times N_h}$ scales a non-Markovian part determined by the hidden state of the last sequence point $\tau - 1$. The vector $\mathcal{B}^g = (\mathcal{B}_h^g) \in \mathbb{R}^{N_h}$ is a learnable bias. It should be stressed that these parameters do neither depend on t nor on τ and are thus optimized once for the complete dataset $\{\vec{X}^t\}$.

127 Note that this mapping is sometimes extended by a contribution to the $g_h^t[\tau]$ from
 128 the past cell state $\vec{C}^t[\tau - 1]$ (XXX?). Nevertheless, this mechanism called “peeping”
 129 is not further considered in this work.

130 For the sake of brevity, we write the mapping more compactly in matrix-vector form
 131 as

$$\vec{g}^t[\tau] = \mathcal{M}^g \vec{X}^t[\tau] + \mathcal{N}^g \vec{H}^t[\tau - 1] + \vec{\mathcal{B}}^g \quad (1)$$

132 Second, the actual gate vectors are computed by the core equations of the LSTM as
 133 proposed by (XXX?):

$$\begin{aligned} \vec{I}^t[\tau] &= \sigma(\vec{i}^t[\tau]) \\ \vec{F}^t[\tau] &= \sigma(\vec{f}^t[\tau]) \\ \vec{O}^t[\tau] &= \sigma(\vec{o}^t[\tau]) \\ \vec{C}^t[\tau] &= \vec{F}^t[\tau] \circ \vec{C}^t[\tau - 1] + \vec{I}^t[\tau] \circ \tanh(\vec{c}^t[\tau]) \\ \vec{H}^t[\tau] &= \vec{O}^t[\tau] \circ \tanh(\vec{C}^t[\tau]) , \end{aligned} \quad (2)$$

134 where σ denotes the logistic sigmoid function, \tanh is the hyperbolic tangent and
 135 the \circ stands for the Hadamard product (all applied in an element-wise fashion to the
 136 vectors). In the last two equations the role of the input, forget and output gates as
 137 described above become apparent.

138 The third step in a single layer LSTM (as employed for the work presented here)
 139 is then to provide the output of the current LSTM cell, i.e. $\vec{H}^t[\tau]$ and $\vec{C}^t[\tau]$, to
 140 the subsequent LSTM cell that processes the next element $\vec{X}^t[\tau + 1]$ of the input
 141 sequence.

142 The full action of the LSTM network up to the end of the sequence can therefore be
 143 written as a nested function call

$$(\vec{H}^t[N_\tau], \vec{C}^t[N_\tau]) = L(\vec{X}^t[N_\tau], L(\vec{X}^t[N_\tau - 1], \dots L(\vec{X}^t[1], (\vec{H}^t[0], \vec{C}^t[0]) \dots)) ,$$

144 where L represents Eqs. (Equation 1, Equation 2) and the initial conditions are
 145 usually chosen as $\vec{H}^t[0] = \vec{C}^t[0] = 0$.

146 The final output $\vec{H}^t[N_\tau]$ and $\vec{C}^t[N_\tau]$ encode information on the full input sequence
 147 ending at time t . This information has to be decoded to obtain useful information
 148 via an appropriate subsequent layer, as it is described in the Sec. (XXX?).

149 A significant advantage of this architecture is the memory cell’s ability to retain gra-
 150 dients. This mechanism addresses the vanishing gradient problem, where, as input
 151 sequences elongate, the influence of initial stages becomes harder to capture, causing
 152 gradients of early input points to approach zero. The LSTM’s activation function,
 153 inherently recurrent, mirrors the identity function with a consistent derivative of 1.0,
 154 ensuring the gradient remains stable throughout backpropagation.

155 2.5 ConvLSTM network

156 Although the plain LSTM has high performance in handling temporal sequences of
 157 point-like quantities it is not designed to recognize spatial features in sequences of
 158 two-dimensional maps. To address this limitation we use a convLSTM architecture.

159 In this kind of network the elements of the input sequence are given as (kind
 160 of) vector fields $\vec{X}^t[\tau] = (X_k^t[x, y, \tau]) \in \mathbb{R}^{N_k \times (N_x \times N_y)}$, where $x \in [1, N_x]$ and
 161 $y \in [1, N_y]$ run over the horizontal and vertical dimensions of the map, respec-
 162 tively. In order to enable the “learning” of spatial patterns, the free parameters
 163 of the network are replaced by two-dimensional convolutional kernels $\mathcal{M}^g =$
 164 $(\mathcal{M}_{hk}^g[\xi, \eta]) \in \mathbb{R}^{(N_h \times N_k) \times (N_\xi \times N_\eta)}$ and $\mathcal{N}^g = (\mathcal{N}_{hh'}^g[\xi, \eta]) \in \mathbb{R}^{(N_h \times N_h) \times (N_\xi \times N_\eta)}$.

165 The width and the height of the kernels are given by N_ξ and N_η , respectively and
 166 $\xi \in [-(N_\xi - 1)/2, (N_\xi - 1)/2]$, $\eta \in [-(N_\eta - 1)/2, (N_\eta - 1)/2]$, where we assume odd
 167 numbers for the kernel sizes.

168 The mapping from the input quantities to the gates is then given by a convolution
 169 with these kernels

$$g_h^t[x, y, \tau] = \mathcal{M}_{hk}^g[\xi, \eta] X_k^t[x - \xi, y - \eta, \tau] + \mathcal{N}_{hh'}^g[\xi, \eta] H_{h'}^t[x - \xi, y - \eta, \tau - 1] + \mathcal{B}_h^g[x, y].$$

170 It becomes immediately apparent that in case of the convLSTM, the bias, gate and
 171 state vectors must become vector fields ($\in \mathbb{R}^{N_h \times (N_x \times N_y)}$) as well. We can write this
 172 mapping in the same fashion as Eq. (Equation 1) but with replacing the normal
 173 matrix-vector multiplication by a convolution (denoted with $*$), i.e.

$$\vec{g}^t[\tau] = \mathcal{M}^g * \vec{X}^t[\tau] + \mathcal{N}^g * \vec{H}^t[\tau - 1] + \vec{\mathcal{B}}^g$$

174 The subsequent processing of the $\vec{g}^t[\tau]$ remains symbolically the same as presented
 175 in Eq. (Equation 2) but with all appearing quantities meaning standing now for
 176 vector fields instead of simple vectors.

177 In summary, the ConvLSTM excels at processing tasks that demand a combined
 178 understanding of spatial patterns and temporal sequences in data. It merges the
 179 image-processing capabilities of Convolutional Neural Networks (CNNs) with the
 180 time-series modeling of Long Short-Term Memory (LSTM) networks.

181 2.6 Implemented model architecture

182 The ConvLSTM architectures uses an encoder/decoder structure as discussed in
 183 TODO. To predict all 97 rivers entering the Baltic Sea, we flatten the output and
 184 use fully connected layers to map onto the individual rivers outputs.

185 An overview of the model structure is given below

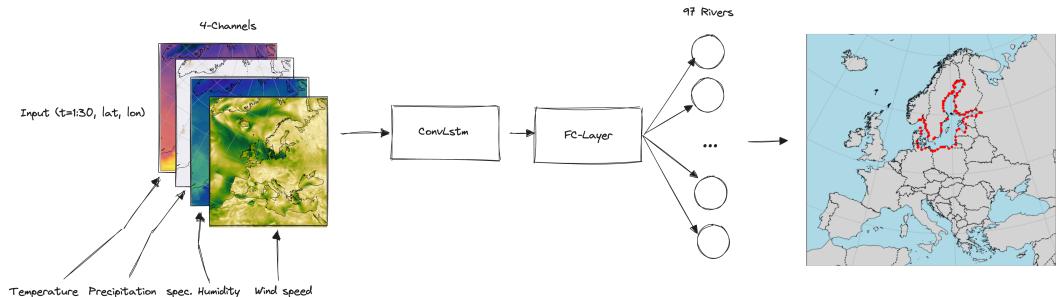


Figure 2: Schematic structure of the convLSTM implementation for river runoff forecasting.

186 For the computation we use the following set of hyper parameters:

Table 1: Hyperparameters

Parameter name	Parameter size
Channel size	4
Num. hidden layer	9
Num. timesteps	30
Conv. Kernelsize	(7,7)

Parameter name	Parameter size
Num. ConvLSTM layers	1
Batch size	50
Learning Rate	1e-3 with CosineAnnealing

187 As input the model receives 30 days of atmospheric surface fields temperature T ,
 188 precipitation P , specific humidity Q and wind speed W , with a daily resolution to
 189 predict the river runoff R at the time step $\Delta t + 1$, which can be summarized as

190
$$R_{\Delta t+1} = f(T_{t-30:t}, P_{t-30:t}, Q_{t-30:t}, W_{t-30:t})$$

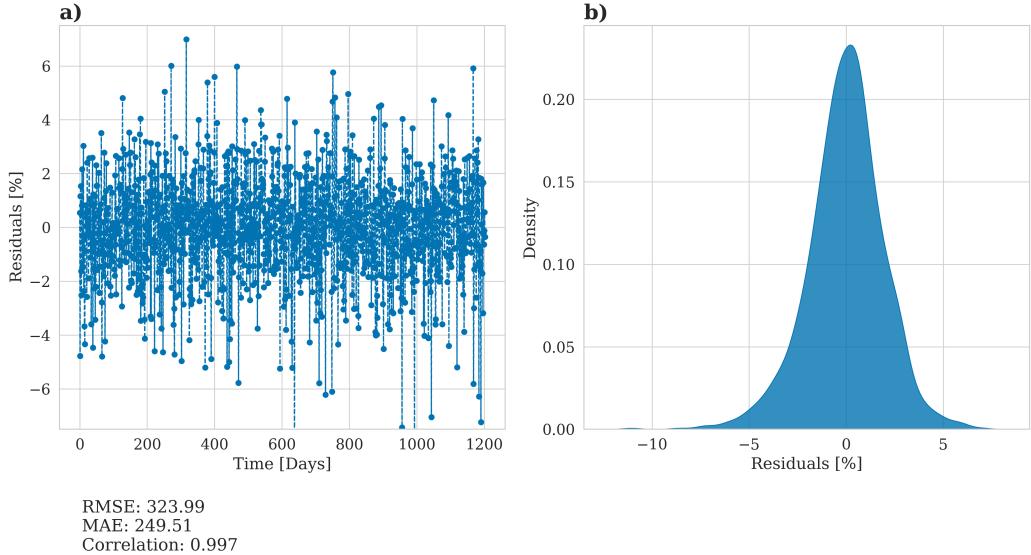
191 with f being a function maps the 30 days of daily atmospheric surface fields data to
 192 the predicted river runoff.

193 The choice of atmospheric fields was based on the implemented river runoff cal-
 194 culation in the atmospheric model COSMO-CLM which uses these four fields to
 195 calculate an river runoff estimate.

196 3 Results

197 The model was trained with daily data for the period 1979 to 2011, as this period
 198 represents the only period of E-HYPE without further bias correction applied to the
 199 runoff to match observations. The data was divided into randomly chosen splits of
 200 80% training data, 10% validation data to evaluate the performance of the model
 201 during training, and 10% training data which is finally used to evaluate the perfor-
 202 mance of the model after training. The model was trained for 400 epochs and the
 203 model weights with the lowest mean squared error during training have been stored.

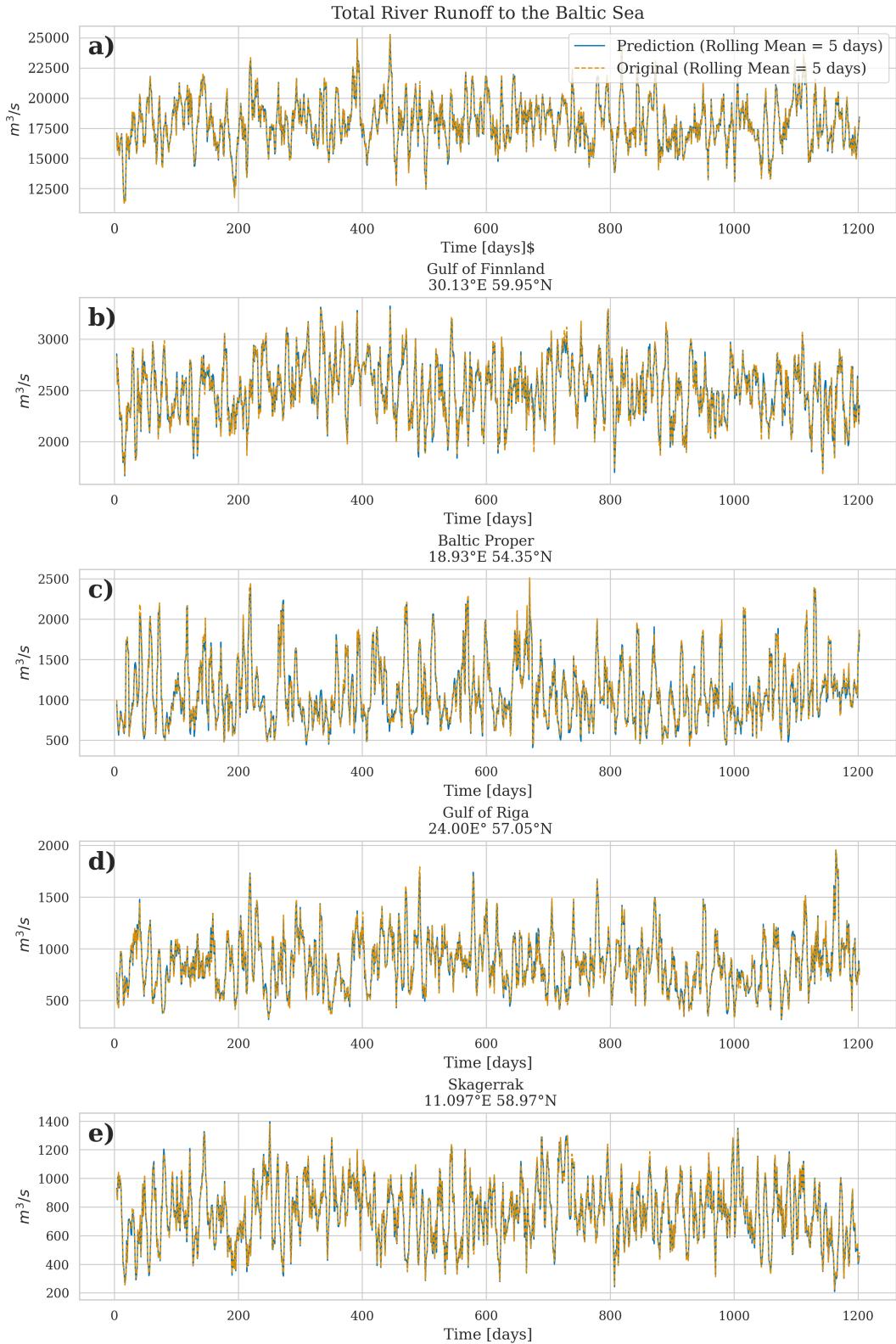
204 The accuracy of the model is displayed in Figure ?@fig-statistical-evaluationNN.
 205 As mention above for evaluation, the test dataset was utilized. The left panel panel
 206 (Figure ?@fig-statistical-evaluationNN a)) illustrates the relative prediction
 207 error in relation to the original E-HYPE data, indicating that, on daily timescales,
 208 the model can predict river runoff with an accuracy of $\pm 5\%$. The overall corre-
 209 lation is 0.997 with the resulting error metrics yielding a RMSE of $323.99 \text{ m}^3/\text{s}$ and
 210 MAE of $249.51 \text{ m}^3/\text{s}$. While the model's performance is satisfactory, the discrep-
 211 ancies between the actual values and the predictions could partly be attributed to
 212 the use of a different atmospheric dataset than the one originally used to drive the
 213 E-HYPE model. However, by applying a rolling mean with a 5-day window, the
 214 prediction error is reduced to less than 1%, which is acceptable for the purposes of
 215 climate modeling. Lastly, the right panel demonstrates that the distribution of resid-
 216 uals follows a Gaussian shape, suggesting that our model does not exhibit bias by
 217 systematically over or underestimating river runoff values.



218

219

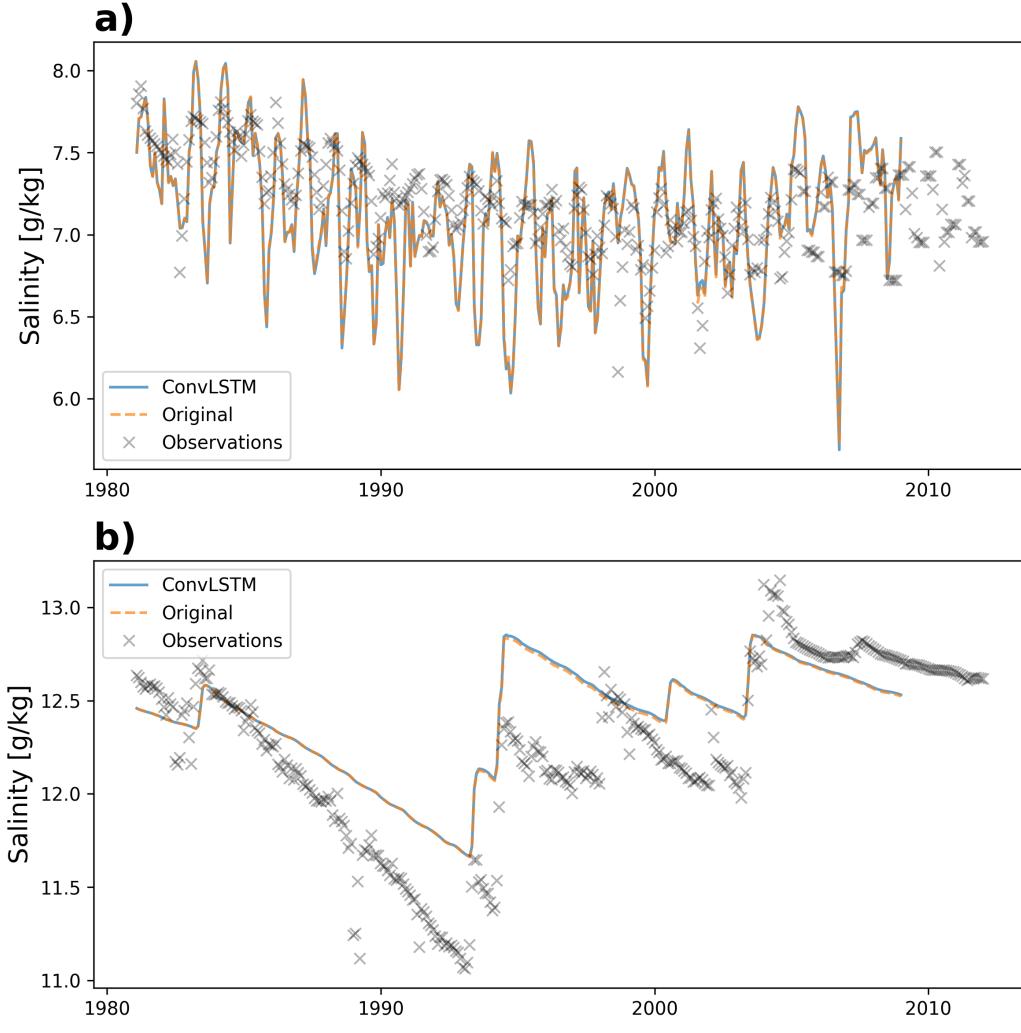
In the following we will now address the overall performance of the total river runoff while also zooming in on the four largest rivers entering the Baltic Sea. @fig-PerformanceNeuralNetworkRunoff shows the predicted and the original river runoff using the test dataset. The predicted total river runoff for the Baltic Sea is closely matching the original data (@fig-PerformanceNeuralNetworkRunoff a). Zooming in on the largest individual rivers (@fig-PerformanceNeuralNetworkRunoff b-e) it can be seen that that also the prediction of the individual rivers is close to the original data.



227

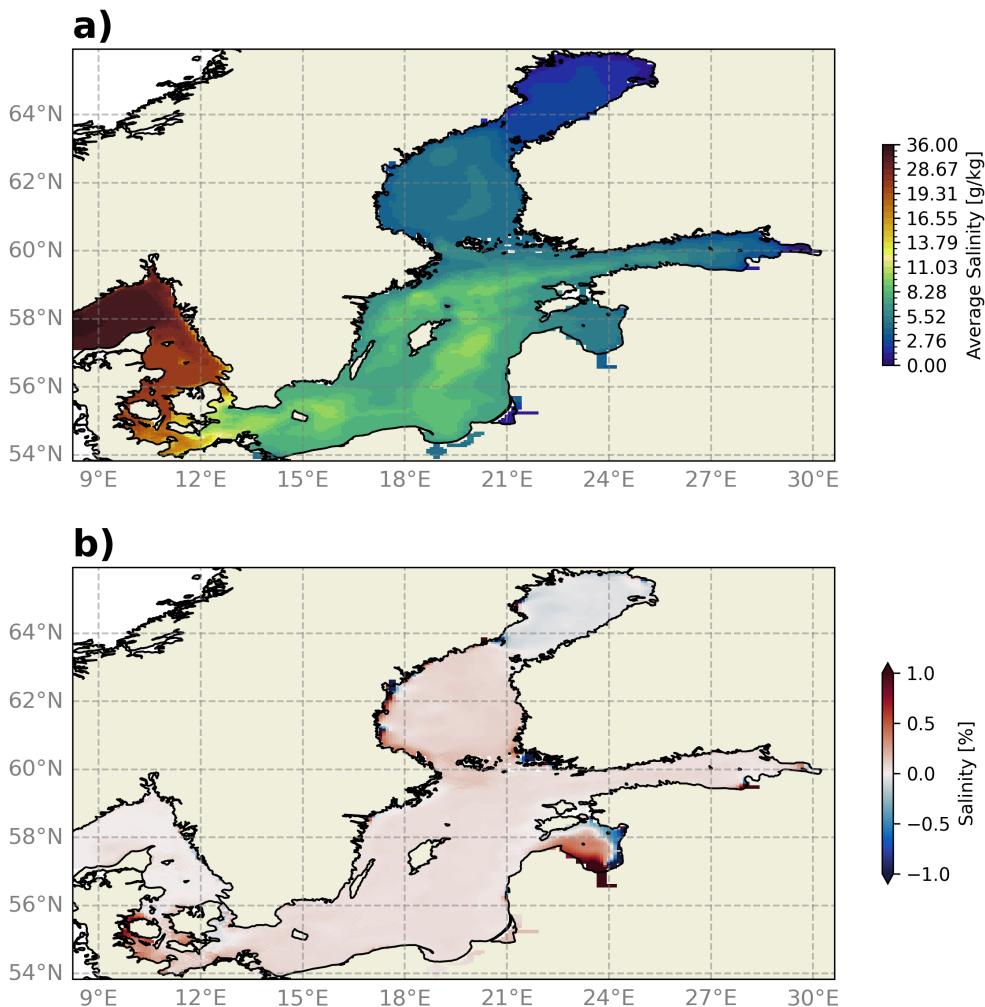
228 Lastly, we evaluated the performance of the runoff model by incorporating the pre-
 229 dicted river runoff as runoff forcing into the ocean model MOM5. This provides a
 230 robust validation of the runoff model against more complex real world conditions.
 231 This allows us to ensure that the predictions accurately reflect the impact of the
 232 river discharge on the ocean dynamics, validating the temporal and spatial variabil-

233 ity of the the river discharge. ?@fig-by15 shows the salinity comparison between
 234 the original E-HYPE river runoff and the predicted river runoff at BY15 - a cen-
 235 tral stations in the Baltic Sea. It can be seen that the model simulation using the
 236 predicted river runoff by the ConvLSTM is closely mirroring the control simulation.
 237 The upper panel (?@fig-by15 a) shows the surface salinity, representing the high-
 238 frequency variations in the salinity, which is heavily affected by river runoff. The
 239 lower panel (?@fig-by15 b) shows the bottom salinity which can be viewed as a
 240 low-pass filter in the Baltic Sea, which is also closely mirrored by the ConvLSTM
 241 predictions.



242

243 The final evaluation of the ConvLSTM model concentrates on the spatial accuracy
 244 of river runoff predictions. Figure xxx a) shows the vertically averaged salinity for
 245 the period 1981 to 2011 in the reference simulation. It highlights the strong horizon-
 246 tal gradients and complex topographic features in the Baltic Sea, as evidenced by
 247 salinity variations in deeper waters, captured by the vertical integration. Figure 4b
 248 compares these results by showing the percentage difference in vertically averaged
 249 salinity between the ConvLSTM simulation and the reference simulation. Overall,
 250 the differences remain below 1%, except near a river mouth in the Gulf of Riga
 251 (22-24°E, 56.5-58.5°N for orientation), where the difference is approximately 1%.



252

253

4 Acknowledgments

Phasellus interdum tincidunt ex, a euismod massa pulvinar at. Ut fringilla ut nisi nec volutpat. Morbi imperdiet congue tincidunt. Vivamus eget rutrum purus. Etiam et pretium justo. Donec et egestas sem. Donec molestie ex sit amet viverra egestas. Nullam justo nulla, fringilla at iaculis in, posuere non mauris. Ut eget imperdiet elit.

258

5 Open research

Phasellus interdum tincidunt ex, a euismod massa pulvinar at. Ut fringilla ut nisi nec volutpat. Morbi imperdiet congue tincidunt. Vivamus eget rutrum purus. Etiam et pretium justo. Donec et egestas sem. Donec molestie ex sit amet viverra egestas. Nullam justo nulla, fringilla at iaculis in, posuere non mauris. Ut eget imperdiet elit.

264

Fang, W., Huang, S., Ren, K., Huang, Q., Huang, G., Cheng, G., & Li, K. (2019).

265

References Examining the applicability of different sampling techniques in the development of decomposition-based streamflow forecasting models. *Journal of Hydrology*, 568, 534–550. <https://doi.org/10.1016/j.jhydrol.2018.11.020>

266

Hagemann, S., Stacke, T., & Ho-Hagemann, H. T. M. (2020). High Resolution Discharge Simulations Over Europe and the Baltic Sea Catchment. *Frontiers in Earth Science*, 8. <https://doi.org/10.3389/feart.2020.00012>

267

268

269

270

- 271 Tan, Q.-F., Lei, X.-H., Wang, X., Wang, H., Wen, X., Ji, Y., & Kang, A.-Q. (2018).
272 An adaptive middle and long-term runoff forecast model using EEMD-ANN hy-
273 brid approach. *Journal of Hydrology*, 567, 767–780. <https://doi.org/10.1016/j.jhydrol.2018.01.015>
- 274 Yang, Z., Liu, P., Cheng, L., Wang, H., Ming, B., & Gong, W. (2018). Deriving
275 operating rules for a large-scale hydro-photovoltaic power system using im-
276 plicit stochastic optimization. *Journal of Cleaner Production*, 195, 562–572.
277 <https://doi.org/10.1016/j.jclepro.2018.05.154>
- 278