STREAMFLOW FORECASTING IN THE EBRO RIVER BASIN USING MACHINE LEARNING AND A PHYSICAL MASS CONSTRAINT

Inmaculada C. González-Planet[1], Carmelo Juez[1]

[1] Instituto Pirenaico de Ecología, IPE-CSIC, Zaragoza, Spain

ic.planet@ipe.csic.es

This presentation participates in OSPP — Outstanding Student & PhD candidate Presentation contest

QR code to abstract

European Geosciences Union 2025

Vienna, Austria & Online 27 April–2 May 2025

Session ITS 1.1 / CL 0.9

Introduction

The increasing influence of climate variability demands robust and flexible forecasting approaches. Streamflow forecasting is essential for water resources management, agricultural planning, flood risk prevention and mitigation, hydropower generation and the preservation of freshwater ecosystems. Although numerous methods have been developed for streamflow prediction, it remains a challenge due to the complex and nonlinear nature of hydrological systems.

The use of machine learning (ML) in hydrology has gained traction due to its ability to provide alternative or complementary approaches to traditional process-based modelling. These models identify numerical patterns in time series data without needing to solve conservation equations. This flexibility enables hydrological calculations in areas where data sources are incomplete or non-existent.

In this study, a Long Short-Term Memory (LSTM) neural network has been designed (Hochreiter & Schmidhuber, 1997) and its mass-conserving variant, the Mass-Conserving LSTM (MC-LSTM) neural network (Hoedt et al., 2021), to learn sequential relationships between atmospheric, climatic and geographic features and daily streamflow data. The models have been trained and evaluated using observed data from 39 headwater gauging stations in the northern Ebro river basin (Fig 1).

Study area

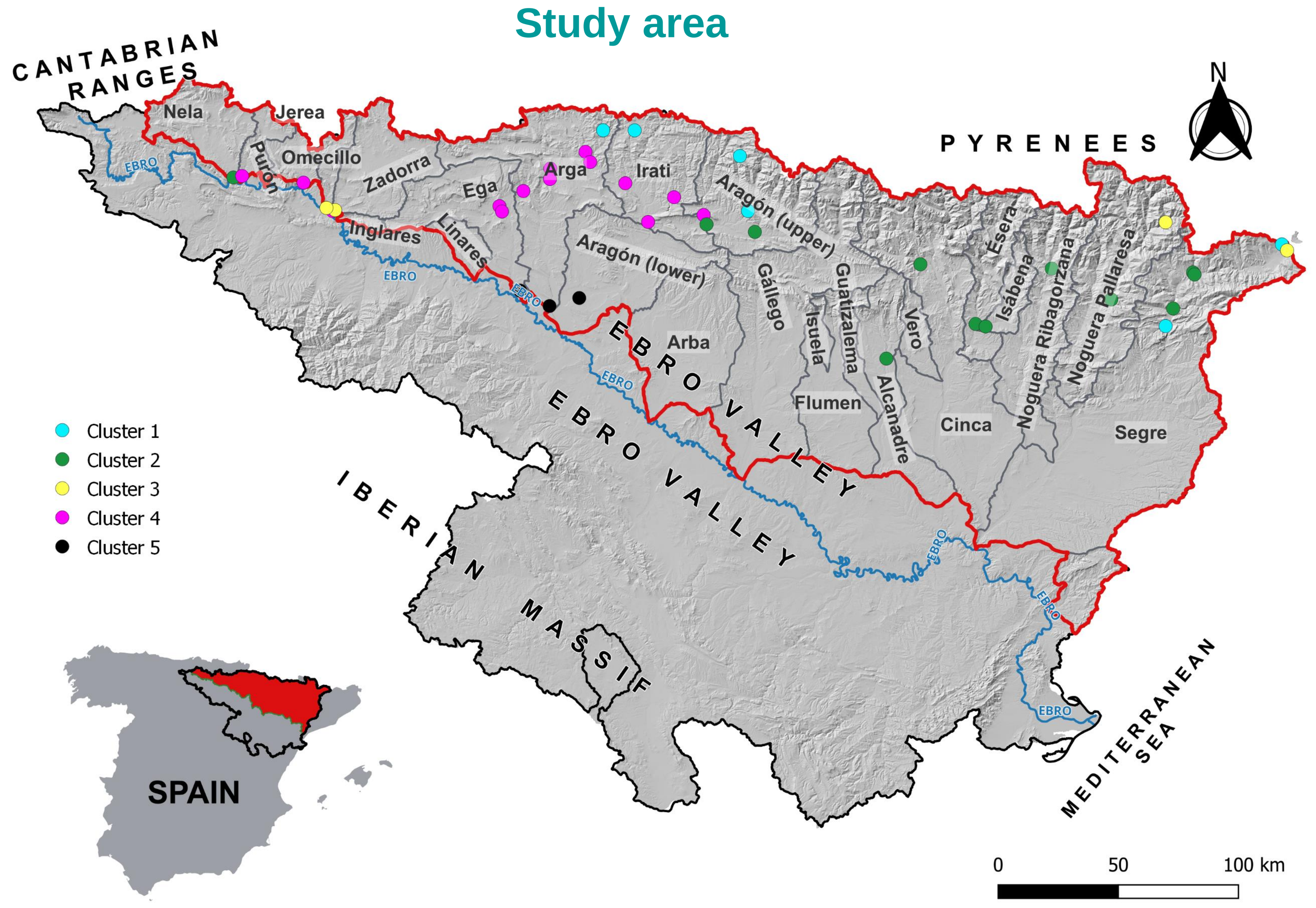


Fig 1. Location of the northern part of the Ebro river basin. Their colours represent a distinct cluster, highlighting regional hydrological similarities.

Methodology

We benchmarked the global outputs of models with respect to the NSE metric. Since NSE metric represents an overall measure of performance, it does not specifically capture the ability of models to predict critical events, such as extreme flood events. Therefore, we designed a complementary metric to evaluate the detection of extreme values based on the RMSE associated with a specific return period.

Error metrics

Global error metric: $NSE = 1 - \frac{\sum_{t=1}^{L}(y^t - \hat{y}^t)^2}{\sum_{t=1}^{L}(y^t - \bar{y})^2}$

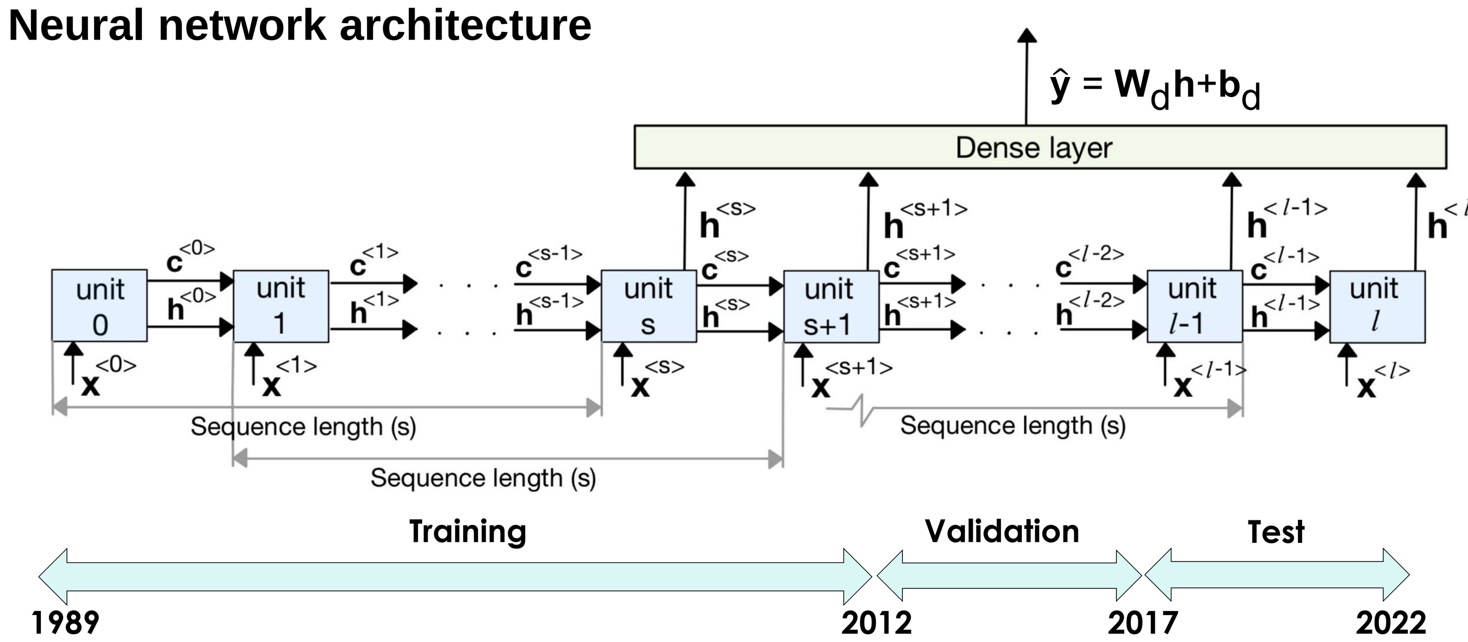
Flood events error metric: $RMSE_{T_x}\left(\frac{m^3}{s}\right) = \sqrt{\frac{\sum (y^t - \hat{y}^t)^2}{l}}$ where $T_{0.5} \le y^t < T_1$ if $x = 0.5$; $T_1 \le y^t < T_2$ if $x = 1$; $y^t \ge T_2$ if $x = 2$

To assess the performance of the models, three different streamflow prediction strategies were developed. The objective is to compare the predictive results under three training configurations: i) TS1: Individual models trained for each gauging station, resulting in as many models as there are gauging stations, ii) TS2: A single regional model trained using data from all gauging stations, and iii) TS3: Cluster-based training, where a model is trained for each group of gauging stations within a clustering scheme (Fig 1), resulting in a total of five different models.

Training scenarios

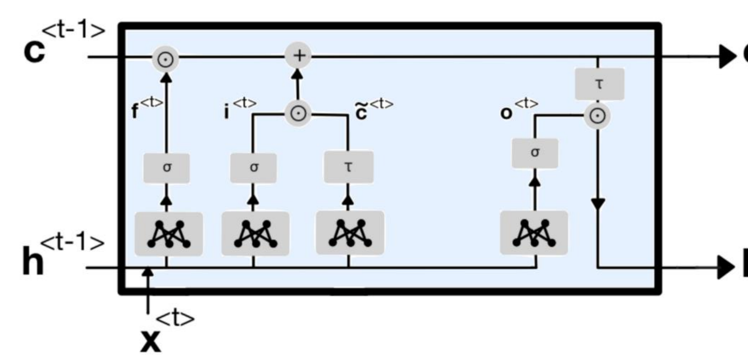
TS1	TS2	TS3
Individual models	Regional model	Cluster-based models

a) Neural network architecture

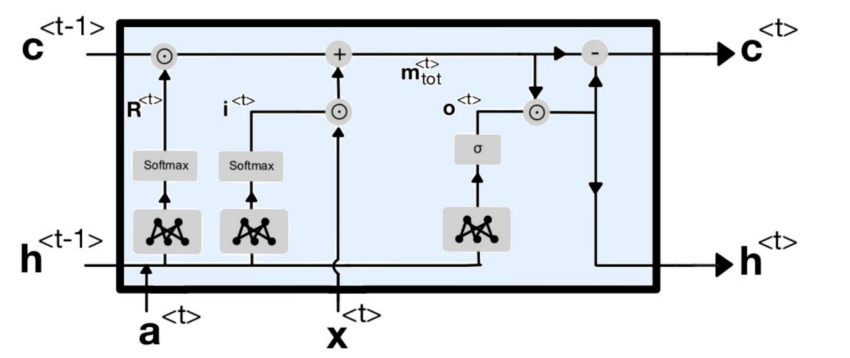


INPUT FEATURES		
Daily dynamic time series data		Static data
Climatic	Atmospheric	Geographic
Precipitation Temperature Air humidity	NAO WEMO SPEI	Drainage area Elevation Land uses

b) The LSTM unit



c) The MC-LSTM unit



Neural network · σ Sigmoid function · τ Hyperbolic tangent function · Softmax Softmax function

+ Addition operation · - Subtraction operation · ⊙ Element-wise multiplication

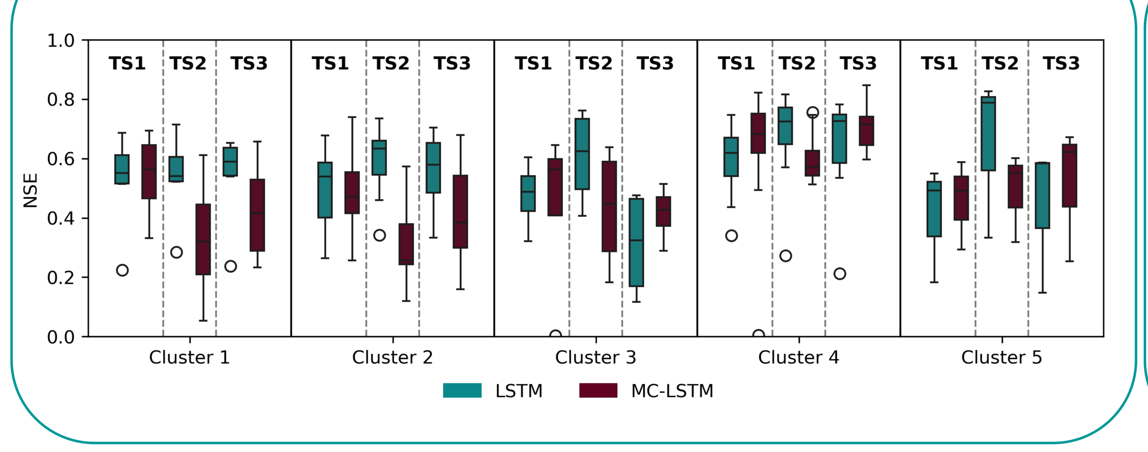
$$
\begin{aligned}
\boldsymbol{f}^t &= \sigma(\boldsymbol{W}_f \cdot [\boldsymbol{h}^{t-1}, \boldsymbol{x}^t] + \boldsymbol{b}_f)\\
\boldsymbol{i}^t &= \sigma(\boldsymbol{W}_i \cdot [\boldsymbol{h}^{t-1}, \boldsymbol{x}^t] + \boldsymbol{b}_i)\\
\tilde{\boldsymbol{c}}^t &= \tau(\boldsymbol{W}_{\tilde{c}} \cdot [\boldsymbol{h}^{t-1}, \boldsymbol{x}^t] + \boldsymbol{b}_{\tilde{c}})\\
\boldsymbol{o}^t &= \sigma(\boldsymbol{W}_o \cdot [\boldsymbol{h}^{t-1}, \boldsymbol{x}^t] + \boldsymbol{b}_o)\\
\boldsymbol{c}^t &= \tilde{\boldsymbol{c}}^t \odot \boldsymbol{i}^t + \boldsymbol{c}^{t-1} \odot \boldsymbol{f}^t\\
\boldsymbol{h}^t &= \tau(\boldsymbol{c}^t) \odot \boldsymbol{o}^t
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{R}^t &= softmax(\boldsymbol{B}_r)\\
\boldsymbol{i}^t &= softmax\left(\boldsymbol{W}_i \cdot \left[\boldsymbol{a}^t, \frac{\boldsymbol{c}^{t-1}}{\|\boldsymbol{c}^{t-1}\|_1}\right] + \boldsymbol{b}_i\right)\\
\boldsymbol{o}^t &= \sigma\left(\boldsymbol{W}_o \cdot \left[\boldsymbol{a}^t, \frac{\boldsymbol{c}^{t-1}}{\|\boldsymbol{c}^{t-1}\|_1}\right] + \boldsymbol{b}_o\right)\\
\boldsymbol{m}_{tot}^t &= \boldsymbol{R}^t \cdot \boldsymbol{c}^{t-1} + \boldsymbol{i}^t \cdot \boldsymbol{x}_m^t\\
\boldsymbol{c}^t &= (1 - \boldsymbol{o}^t) \odot \boldsymbol{m}_{tot}^t\\
\boldsymbol{h}^t &= \boldsymbol{o}^t \odot \boldsymbol{m}_{tot}^t
\end{aligned}
$$

Fig 2. a) Architecture of a sequential model. Each output is associated with s, the sequence length. l represents the total number of desired time steps. The inputs to the dense layer are the final hidden states of each sequence length, where are transformed into the output vector. The length of the output vector depends on both the number of time steps and the sequence length $\hat{\boldsymbol{y}} = [\hat{y}_s, \hat{y}_{s+1}, \dots, \hat{y}_l]$. **b)** The LSTM unit architecture. t and $t-1$ is the current and previous time step of time series; $\boldsymbol{h}^t$ and $\boldsymbol{h}^{t-1}$ is current and previous hidden state; $\boldsymbol{c}^t$ and $\boldsymbol{c}^{t-1}$ is current and previous memory cell; $\boldsymbol{x}^t$ is the input vector that contains features used at t time step; $\boldsymbol{f}^t$, $\boldsymbol{i}^t$ and $\boldsymbol{o}^t$ are forget, input and output gates; $\tilde{\boldsymbol{c}}^t$ is the update gate. **c)** The MC-LSTM unit architecture. $\boldsymbol{a}^t$ is the current auxiliary input vector, $\boldsymbol{x}_m^t$ is the current mass input vector, $\boldsymbol{R}^t$ is the redistribution matrix in time step t, and $\boldsymbol{m}_{tot}^t$ is the total mass vector.

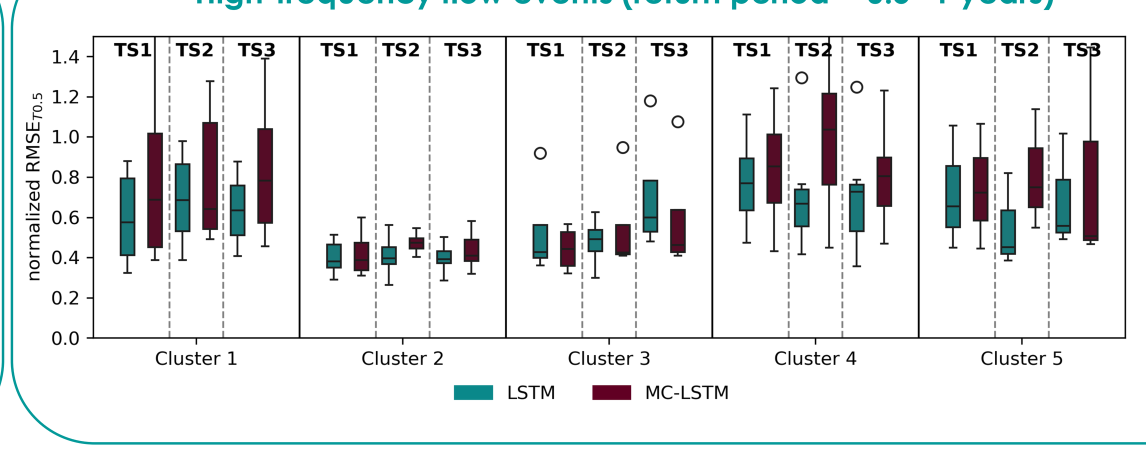
Results

Global error metric

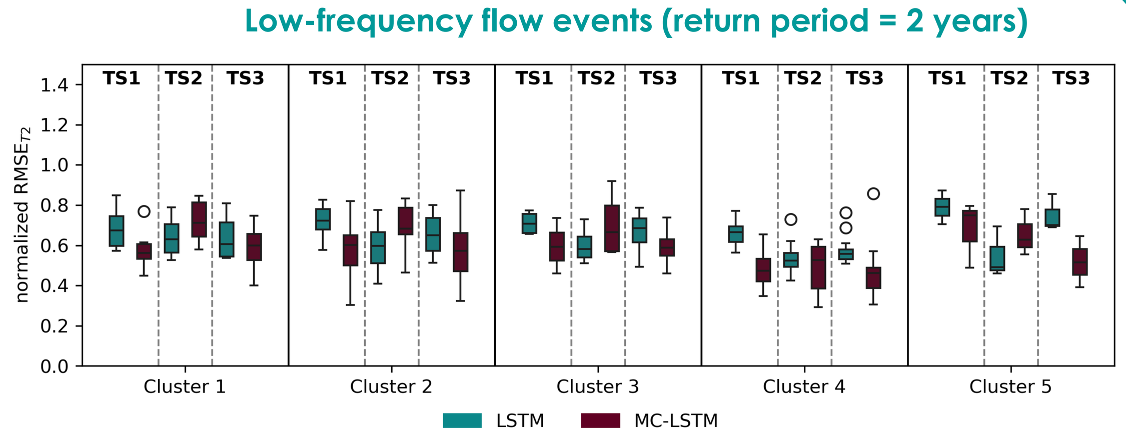


Flood events error metric

High-frequency flow events (return period = 0.5 -1 years)



Low-frequency flow events (return period = 2 years)



Conclusions

The LSTM architecture demonstrates the best performance, with the TS2 scenario achieving the highest median NSE across the three configurations, reaching a value of 0.66.

In relation to the magnitude of flow events, the LSTM model performs more reliably in capturing moderate flow events, those that occur more frequently and are better represented in the training data. However, the improved performance of the MC-LSTM model in TS1 and TS3 for high-magnitude, low-frequency events may be attributed to its mass-conserving structure. This physical constraint acts as a regularization mechanism, enhancing the model's ability to generalize when predicting events that are sparsely represented in the training dataset.

For future work, it is proposed to further investigate the MC-LSTM architecture to gain a deeper understanding of the underlying reasons behind the results obtained.

Acknowledgments

This work is funded by the European Research Council (ERC) through the Horizon Europe 2021 Starting Grant program under REAgrant agreement number 101039181 - SED@ HEAD.

References

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. Neural computation, 9, 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

Hoedt, P.-J., Kratzert, F., Klotz, D., Halmich, C., Holzleitner, M., Nearing, G., Hochreiter, S., & Klambauer, G. (2021). MC-LSTM: Mass-Conserving LSTM (No. arXiv:2101.05186). arXiv. http://arxiv.org/abs/2101.05186