

Article

Not peer-reviewed version

Investigating the Performance of the Informer Model for Streamflow Forecasting

[Nikos Tepetidis](#), [Demetris Koutsoyiannis](#)*, [Theano Iliopoulou](#), [Panayiotis Dimitriadis](#)

Posted Date: 16 August 2024

doi: 10.20944/preprints202408.1084.v1

Keywords: flood; river streamflow forecasting; long sequence time series forecasting (LSTF); deep learning; transformers; attention mechanism; Informer model



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Investigating the Performance of the Informer Model for Streamflow Forecasting

Nikos Tepetidis, Demetris Koutsoyiannis * , Theano Iliopoulou  and Panayiotis Dimitriadis 

Department of Water Resources and Environmental Engineering, School of Civil Engineering, National Technical University of Athens, 15780 Zographou, Greece; nikostepe191201@gmail.com (N.T.); tiliopoulou@hydro.ntua.gr (T.I.); pandim@itia.ntua.gr (P.D.)

* Correspondence: dk@itia.ntua.gr

Abstract: Recent studies have shown the potential of transformer-based neural networks in increasing prediction capacity. However, classical transformers present several problems such as computational time complexity and high memory requirements, which make Long Sequence Time-Series Forecasting (LSTF) challenging. The contribution to the prediction of time series of flood events using deep learning techniques is examined, with a particular focus on evaluating the performance of the Informer model (a particular implementation of transformer architecture), which attempts to address the previous issues. This study explores the predictive capabilities of the Informer model compared to statistical methods, stochastic models and traditional deep neural networks. The accuracy, efficiency as well as the limits of the approaches are demonstrated via numerical benchmarks relating to real river streamflow applications. Using daily flow data from the River Test in England as the main case study, we conduct a rigorous evaluation of the Informer efficacy in capturing the complex temporal dependencies inherent in flood time series. Among other things, the present work extends its analysis to encompass diverse time series datasets from various locations (>100) in the United Kingdom, providing insights into the generalizability of the Informer. The results highlight the significant superiority of the Informer model over established forecasting methods, especially regarding the LSTF problem. Furthermore, it is observed that the structure of time series, as expressed by climacogram, affects the performance of the Informer network.

Keywords: flood; river streamflow forecasting; long sequence time series forecasting (LSTF); deep learning; transformers; attention mechanism; Informer model

1. Introduction

Flood events refer to the natural phenomena characterized by the overflow or inundation of land areas that are typically dry, caused by excessive accumulation of water. These events can occur due to various factors such as heavy rainfall, snowmelt, dam failure, or coastal surges. Floods are of huge importance worldwide as natural disasters that affect different regions, leaving a profound impact on societies [1]. They transcend geographical boundaries and affect millions of lives around the world, making them a critical area of study and research. Understanding the impacts and mitigating the risks associated with flood events is of paramount importance to safeguard human life and minimise socio-economic and environmental damage. They can result in loss of life, property damage, displacement of communities, disruption of essential services, and long-term ecological consequences. The severity and frequency of flooding events vary globally, making accurate forecasting and alertness crucial for effective disaster management and mitigation strategies. In the context of the above, the accurate prediction of daily river streamflow, which constitutes an important factor in flooding, is essential.

Precise streamflow forecasting is particularly needed in agricultural water management, because it affects irrigation scheduling, mitigation of drought and flood, and many other actions that ensure agriculture stability and optimize crop yields. On the other hand, timely and accurate forecasts can reduce the impact of drought, allowing for proactive water management strategies through adjustments in irrigation and water-saving plans ahead of the dry seasons [2]. Likewise, streamflow predictions are very vital in flood prevention and management, thus protecting farmland from flooding and associated losses. Hence, this pre-emptive approach provides a robust tool for agriculture water managers to make informed decisions for improving the overall efficiency of water use in agriculture.

Various techniques are employed to achieve accurate and reliable results for estimating river streamflow, most of which are either physically based models or data-driven. Physically based models for streamflow prediction depend on specific hydrological hypotheses and necessitate extensive hydrological data for effective calibration [3]. They represent hydrological systems very truthfully and show what is happening with the processes involved. They also prove particularly beneficial when a deep understanding of the physical processes is essential, such as assessing the impact of land use changes or climatic variations on streamflow [4]. Beyond data constraints [5], these models often require complex parameterization, a process prone to challenges and imprecision. Unfortunately, the calibration of these models is resource-intensive and time-consuming. As a result, there is a growing interest in enhancing data-driven models for streamflow prediction, which are more efficient and cost-effective.

For time series prediction of river streamflow, advanced approaches like statistical models, autoregressive integrated moving average (ARIMA) [6], and seasonal decomposition methods are widely used. These statistical models leverage historical data to forecast future flow values, considering seasonality and trends. Moreover, machine learning algorithms, including k-Nearest Neighbors (KNN) [7], Artificial Neural Networks (ANN), Light Gradient Boosting Machine Regressor (LGBM), Linear Regression (LR), Multilayer Perceptron (MLP), Random Forest (RF), Stochastic Gradient Descent (SGD), Extreme Gradient Boosting Regression (XGBoost), Support Vector Regression (SVR) have been widely used as well [4]. There is also great interest in deep learning methods, like Recurrent Neural Networks (RNN), Long Short-Term Memory (LSTM) [8] networks and Convolutional Neural Networks (CNN) [9]. Some other models follow a combined methodology utilizing artificial neural networks that obey any given law of physics, known as Physics Informed Neural Networks (PINNs) [10].

In recent years, modern deep learning models like Transformers, have gained popularity (Figure 1) for many tasks, including time series prediction, due to their ability to capture complex temporal long dependencies and non-linear patterns. Through an attention mechanism, transformers grants a complete and interrelated view of the entire time series being analyzed. These methods could play a crucial part in accurate forecasting of river streamflow and thus empowering water resource managers to make informed decisions for flood management, drought mitigation, and sustainable water allocation.

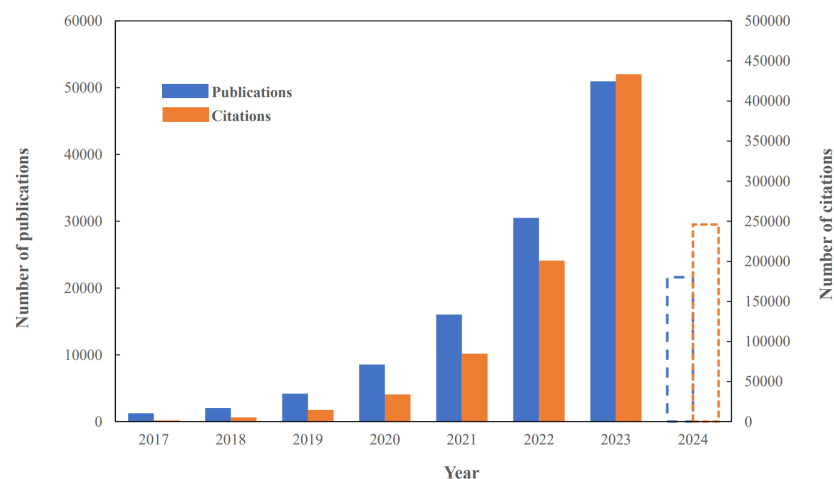


Figure 1. The number of publications and citations per year from 2017 (until 2-06-2024), related to Transformers networks (Data were obtained from Dimensions AI [11], using the keywords: "transformer model", "attention mechanism", "self-attention in neural networks").

1.1. Literature Review

1.1.1. Traditional methods

Time series forecasting has attracted great scientific interest among researchers of many disciplines and for the analysis of various tasks, including the area of hydrology and streamflow prediction. Many algorithms have been used over the years to solve the problem of forecasting, from classical statistical methods to advanced deep learning algorithms. In the book of Box et al. [12], there are several statistical methods and techniques used to analyse and forecast time series data, including the Autoregressive Integrated Moving Average (ARIMA) model [13]. This model follows the Markov process and creates an autoregressive model for recursive sequential forecasting. Exponential Smoothing [14] is used to examine the practical application for handling trend, seasonality, and other components of time series data. Similar models have been applied for streamflow forecasting problem for the last 50 years, like Moving Average (MA), Auto Regressive Moving Average (ARMA), Linear Regression (LR), and Multiple Linear Regression (MLR) [15–17].

Some limitations of these traditional techniques include the frequent assumption that the time series data are linear and stationary. Models such as ARIMA and exponential smoothing have limited ability to approximate complex patterns of time series data, such as anomalies, abrupt changes, slope changes, and non-linear relationships. Another challenge of traditional models is the limitations they may have when processing large-scale time series data or high-dimensional data [18]. As a result, researchers have focused on crafting models designed to mitigate the limitations inherent in conventional models, e.g., data-driven approaches in the broader domain of Artificial Intelligence (AI).

1.1.2. Machine Learning Approaches

In the realm of streamflow forecasting, various machine learning techniques have been used in the literature. Among these, Artificial Neural Networks (ANN) [19] have gained prominence for their ability to model complex relationships within streamflow data. Support Vector Machines (SVM) [20] have also been employed, leveraging their proficiency in handling high-dimensional datasets and capturing nonlinear patterns. Additionally, Random Forests [21] have demonstrated effectiveness, particularly in mitigating issues related to overfitting and enhancing prediction accuracy. Gradient Boosting Machines (GBMs) like the extreme gradient boosting regression (XGBoost) [22], which basically is an optimization of the gradient boosting method for decision tree models, has gained popularity in streamflow prediction [23]. Recent studies demonstrate a growing role that artificial intelligence techniques can also play in improving the accuracy and efficiency of water resource management [24] within agricultural settings. These machine learning approaches offer diverse methodologies for analysing and forecasting flood events, contributing valuable insights to the field and addressing the challenges associated with traditional hydrological models. As researchers dive into the literature, the comparative strengths and weaknesses of these techniques become apparent.

1.1.3. Deep Learning Approaches

More recently there is also great interest in deep learning methods in streamflow forecasting. Commencing with Recurrent Neural Networks (RNNs) and extending to Long Short-Term Memory (LSTM) [25] networks – a distinct subtype of RNNs as well as Gated Recurrent Units (GRUs) [26], capable at capturing temporal relationships in streamflow data and generating accurate predictions. Convolutional Neural Networks (CNNs) [27] can extract critical features from numerous inputs with its convolutions. Likewise, big companies like Amazon and Facebook have gained the opportunity to create their own forecasting model, i.e., DeepAR [28] and Prophet [29] respectively. In [30] researchers implemented Deep Autoregressive (DeepAR), a methodology based on iterative networks such as LSTMs producing probabilistic forecasting, for multi-step-ahead flood probability prediction. On the other side Prophet's core idea is to model time series data as a combination of trend, seasonality and

noise components and its performance is evaluated for daily and monthly streamflow forecasting both in short and long term horizon [21,31].

The year 2017 marked a significant milestone in the field of deep learning with the introduction of the Transformer model, as documented in the relevant work [32]. This groundbreaking model introduced the concept of the attention mechanism, empowering the model to selectively focus on different parts of the input sequence for more effective output generation. Initially applied to natural language processing (NLP), particularly in translation, these models have rapidly gained popularity across various fields and tasks, including time series analysis [33]. The key distinguishing features of Transformer models, that set them apart from other deep learning models (LSTM, GRU, CNN, etc.) include their capacity to capture long-range dependencies, their ability to parallelize tasks, and their effective management of large-scale datasets. Transformer-based applications in time series tasks [33–35], highlighting the advantages, limitations and possible future research directions. The increased utilization of models with transformer architecture prompted researchers to swiftly create various transformer-based models, each tailored by research groups to address distinct challenges and requirements in time series forecasting tasks. In fact several models, e.g, Linformer (NLP)[36], Reformer (NLP) [37], Longformer (NLP) [38], SSDNet [39], TFT [40], Autoformer [41], Triformer [42], TCCT [43], Pyraformer [44], TACTiS [45], FEDformer [46], Scaleformer [47] and Crossformer [48] all share common structure but also have unique features and improvements for the task of time series forecasting. However, it is essential to bear in mind that machine and deep learning models have inherent limitations. Successful model construction relies on large amount and high-quality training data, but challenges such as overfitting must be carefully managed.

In the field of streamflow prediction, researchers lately explored the application of certain transformer models [49–51]; however, these endeavors are still in their early stages of development. The Informer model [52] (i.e., a transformer-based model), highlighted in this study, uniquely combines various characteristics and is worth studying. Specifically designed for predicting long sequences in time series data, Informer has been evaluated on diverse datasets, including electrical transformer temperature, electricity consumption, and climate data. They were also applied in tasks such as drought prediction [53], motor bearing vibrations [54], wind energy production [55], as well as in runoff prediction [56]. The model employs a Kullback-Leibler [57] divergence-based method, i.e., a "distance" measure for probability distributions, to simplify the attention matrix, significantly reducing computational complexity.

1.2. Objectives and Structure of the Study

This paper focuses on the problem of streamflow prediction using state-of-the-art deep learning models, notably the transformer-based models, which have gathered substantial scientific attention and are increasingly recognized for their efficacy in time series prediction. These models demonstrate promising results in capturing temporal dependencies within time series data more efficiently. The first objective is to compare four distinct models, a naive Hurst-Kolmogorov stochastic benchmark model (SB), a time series model (ARIMA), an iterative deep learning network (LSTM), and a transformer-based deep learning model (Informer), in the problem of river streamflow forecasting utilizing a single dataset. Subsequently, to further assess and highlight the performance of the most effective model, the best model was exclusively applied in the forecasting analysis across a large dataset comprising more than 100 individual time series of river streamflow within the United Kingdom territory. This focused evaluation aims to provide a comprehensive understanding of the Informer's capabilities and effectiveness in handling a diverse range of temporal patterns within different time series scenarios.

The contributions of the present paper are as follows:

- Applying a novel approach by incorporating a transformer-based model (Informer) into river streamflow forecasting, contributing to the exploration of advanced deep learning techniques in hydrology. Although these methods (i.e., transformers) are quite advanced, their utilization in hydrology has not been widespread.

- Utilizing and comparing four distinct models (SB, ARIMA, LSTM, Informer) for river streamflow forecasting.
- Conducting an in-depth assessment of the Informer model's performance across a dataset featuring 112 individual time series of river streamflow in the United Kingdom.
- Providing valuable insights into the Informer model's capabilities in handling diverse temporal patterns within different time series scenarios. The performance of the Informer model was assessed in comparison with the structure of the time series.

Apart from the present (Section 1), which provides a brief introduction in the importance of the analysis of flooding events and exploring Artificial Intelligence (AI) approaches, there are four more sections. The methods used are described in Section 2, while the data and results of the two different applications are described in Sections 3, 4. In Section 5 further discussions are provided, and Section 6 concludes the outcomes of this study.

2. Materials and Methods

This section introduces four models (SB, ARIMA, LSTM and Informer) applied to time series forecasting of daily streamflow in the River Test and analyses their limitations.

2.1. Stochastic Benchmark (SB)

Establishing a benchmark model is crucial for comparative analysis. For this task, we introduce the Stochastic Benchmark (SB) method as our benchmark model. The stochastic benchmark method provides a fundamental baseline for evaluation, embodying a simplified yet robust approach in hydroclimatic time series forecasting. This method can serve as an alternative to other benchmark methods, such as the average or naïve methods.

This approach is based on the assumption of a Hurst-Kolmogorov (HK) process, characterized by its climacogram. The climacogram [58] is a key tool in stochastic time series analyses of natural phenomena. The term is defined as the graph of the variance of time-averaged stochastic process $\underline{x}(t)$ (assuming stationarity) with respect to time scale k and is denoted by $\gamma(k)$ [59]. The climacogram is proven valuable in identifying long-term variations, dependencies, persistence, and clustering within a process, a characteristic shared with Transformer models. This can be achieved by using the Hurst (H) coefficient, which is equal to half of the slope of the climacogram on a double logarithmic scale graph.

Considering a stochastic process $\underline{x}(t)$ (here we denote stochastic variables, by underlining them) in continuous time t , representing variables like rainfall or river flow, the cumulative process is defined as follows:

$$\underline{X}(t) := \int_0^k \underline{x}(t) dt \quad (1)$$

Then, $\underline{X}(t)/k$ is the time-averaged process. The climacogram $\gamma(k)$ of the process $\underline{x}(t)$ is the variance of the time-averaged process at time scale k [60], i.e.,

$$\gamma(k) := \text{var} \left[\frac{\underline{X}(t)}{k} \right] \quad (2)$$

In Hurst-Kolmogorov (HK) processes, the future depends on the recorded past and the climacogram is expressed in mathematical form as [61]:

$$\gamma(k) = \lambda^2 \left(\frac{\alpha}{k} \right)^{2-2H} \quad (3)$$

where the α and λ are scale parameters with units of time and $[x]$, respectively, while the coefficient H is the Hurst coefficient, and describes the long-term behaviour of the autocorrelation structure, or the persistence of a process [62]. More specifically, based on the Hurst coefficient, three cases can be distinguished as follows:

- $H = 0.5$; the process has random variation (white noise).
- $0.5 < H < 1$; the process is positively correlated. It seems to be the case in most natural processes, where the time series shows long-term persistence or long-term dependence.
- $0 < H < 0.5$; the process is anti-correlated.

As described in Koutsoyiannis [63] in classical statistics, samples are inherently defined as sets of independent and identically distributed (IID) stochastic variables. Classical statistical methods, aiming to estimate future states, leverage all observations within the sample. The above pose challenges when applied to time series data from stochastic processes. This leads to the following question: what is the required number of past conditions to consider when estimating an average that accurately represents the future average over a given period of length κ ?

The estimation of the local future mean at period length κ with respect to the present and the past, i.e.,

$$\underline{\mu}_{\kappa} := \mathbb{E} \left[\frac{1}{\kappa} (\underline{x}_1 + \dots + \underline{x}_{\kappa}) | \underline{x}_0, \underline{x}_{-1}, \dots \right] \quad (4)$$

Assuming that there are a large number of n observations from the present and the past, but only $\nu \leq n$ are selected for estimation [63], then we have:

$$\hat{\underline{\mu}}_{\nu} := \frac{1}{\nu} (\underline{x}_0 + \underline{x}_{-1} + \dots + \underline{x}_{-\nu+1}) \quad (5)$$

while we can answer the above question by finding the ν that minimizes the mean square error:

$$A(\kappa, \nu) := \mathbb{E} \left[(\hat{\underline{\mu}}_{\nu} - \underline{\mu}_{\kappa})^2 \right] \quad (6)$$

As demonstrated in [63] the standardized mean square error can be expressed in terms of the climacogram as:

$$A(\kappa, \nu) = \left(\frac{1}{\kappa} + \frac{1}{\nu} \right) (\kappa\gamma(\kappa) + \nu\gamma(\nu) - (\nu + \kappa)\gamma(\nu + \kappa)) \quad (7)$$

For the case of the Hurst-Kolmogorov process for which the Equation (3) is valid, the value of ν that minimizes the error A is:

$$\nu = \frac{\kappa}{(\max(0, 2.5H - 1.5))^{2.5}} \quad (8)$$

The above equation (8) distinguishes two cases depending on the value of H coefficient. If $H \leq 0.6$, this yields $\nu = \infty$, which means that the future mean estimate is the average of the entire set of n observations (equivalent to average method). However, if $H > 0.6$, then it can be used $\nu < n$ terms for estimating the future average.

In this paper the above methodology is used to create a new, simple and theoretically substantiated benchmark model for comparison with other models.

2.2. Autoregressive Integrated Moving Average (ARIMA)

For many years now, the ARIMA model has been applied as an important statistical method in making forecasts. It combines three basic components: autoregressive (AR), order of integration (I), and moving average (MA). Thus, in some way, it effectively captures the underlying patterns and dynamics within time series data. This model is characterized by three primary parameters:

1. p - the order of autoregressive terms; The autoregressive component (AR) signifies the correlation between an observation and the number of preceding observations. The parameter p refers to the number of lag observations or autoregressive terms in the model.
2. d - degree of differencing; The differencing component (I) is employed to render the time series stationary by eliminating trends or seasonality. The parameter d denotes the number of

differences required to achieve stationarity, i.e., the statistical properties of the series, such as mean and variance, remain constant over time.

3. q - order of moving average terms; The moving average component (MA) considers the impact of past error terms on the current value. It calculates the weighted average of the previous forecast errors. The parameter q represent the size of the moving average window.

Usually the determination of the best set of parameters (p, d, q) is done with a metric, the aim of which is to find the combination of parameters that provides the optimal balance (trade-off) between model complexity and a strong fit to the training data. In the context of the ARIMA model, the Akaike Information Criterion (AIC) [64] metric serves this role, which compares the goodness of fit of different models. Lower AIC values indicating a better balance between fit and model complexity. It is calculated based on the log-likelihood function and can be written as:

$$\text{AIC} = 2k - 2\ln(L) \quad (9)$$

where L is the maximum log-likelihood and k is the number of model parameters.

2.3. Long Short-Term Memory (LSTM)

Long Short-Term Memory (LSTM) networks represent an enhanced version of Recurrent Neural Networks (RNNs) [65], engineered to hold onto information from previous data for extended periods. They are designed for processing sequential data. LSTMs adeptly address the challenge of long-term dependencies through the incorporation of three gating mechanisms and a dedicated memory unit. They were first introduced by Hochreiter and Schmidhuber in 1997 [66] marking a milestone in deep learning. In many cases it is very important to know the previous time steps for a long length, in order to obtain a correct prediction of the outcome in each problem. This type of networks try to solve the "vanishing gradients" problem [67] (i.e., the loss of information during training that occurs when gradients become extremely small and extend further into the past), which remains the main problem in RNN's. At its core, LSTM consists of various components, including gates and cell state (also referred as the memory of the network), which collectively enable it to selectively retain or discard information over time. In a LSTM network there are three gates as depicted in Figure 2, and the back-propagation algorithm is used to train the model. These three gates and the cell state can be described as follows:

1. **Input gate:** decides how valuable the current input is for solving the task, thus choosing the inflow of new information, selectively updating the memory cell. This gate use hidden state at former time step h_{t-1} and current input x_t to update the input gate with the following equation:

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (10)$$

2. **Forget gate:** decides which information from the previous cell state is to be discarded, allowing the network to prioritize relevant information. This is determined by the sigmoid function. Specifically, it takes the previous hidden state h_{t-1} as well as the current input x_t and outputs a number between 0 (meaning this will be skipped) and 1 (meaning this will be utilized), for each number of the cell state c_{t-1} . The output of the forget gate f_t can be expressed as:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \quad (11)$$

3. **Output gate:** regulates the information that the network outputs, refining the data stored in the memory cell. It decides what the next hidden state should be and this is achieved through the output vector h_t . The output of this gate o_t is described by the equation:

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \quad (12)$$

4. **Cell state:** The long-term dependencies are encoded in the cell states and in this way the problem of "vanishing gradients" can be avoided. Cell state c_t at current time step updates the former c_{t-1} by taking into account the candidate cell state \tilde{c}_t , representing the new information that could be added to the cell state, the input vector i_t and the forget vector f_t . Cell states updated values are as follows:

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \quad (13)$$

$$c_t = \tilde{c}_t \odot i_t + c_{t-1} \odot f_t \quad (14)$$

Final output h_t of the LSTM has the form:

$$h_t = o_t \odot \tanh(c_t) \quad (15)$$

To summarize, Equations (10)–(15) express the mathematical calculations that are performed sequentially in the LSTM model, where i_t , f_t , o_t are the input gate, forget gate and output gate vectors respectively, with subscript t indicating the current time step; W_i , W_f , W_o represent weight matrices for connecting input, forget, and output gates with the input x_t , respectively; U_i , U_f , U_o denote weight matrices for connecting input, forget, and output gates with the hidden state h_{t-1} ; b_i , b_f , b_o are input, forget, and output gate bias vectors, respectively; \tilde{c}_t the candidate cell state and c_t the current cell state. Two activation functions is being used with σ and \tanh stands for sigmoid function and hyperbolic tangent function respectively. Finally \odot operation is the element-wise or Hadamard product and $+$ operation is simply pointwise addition.

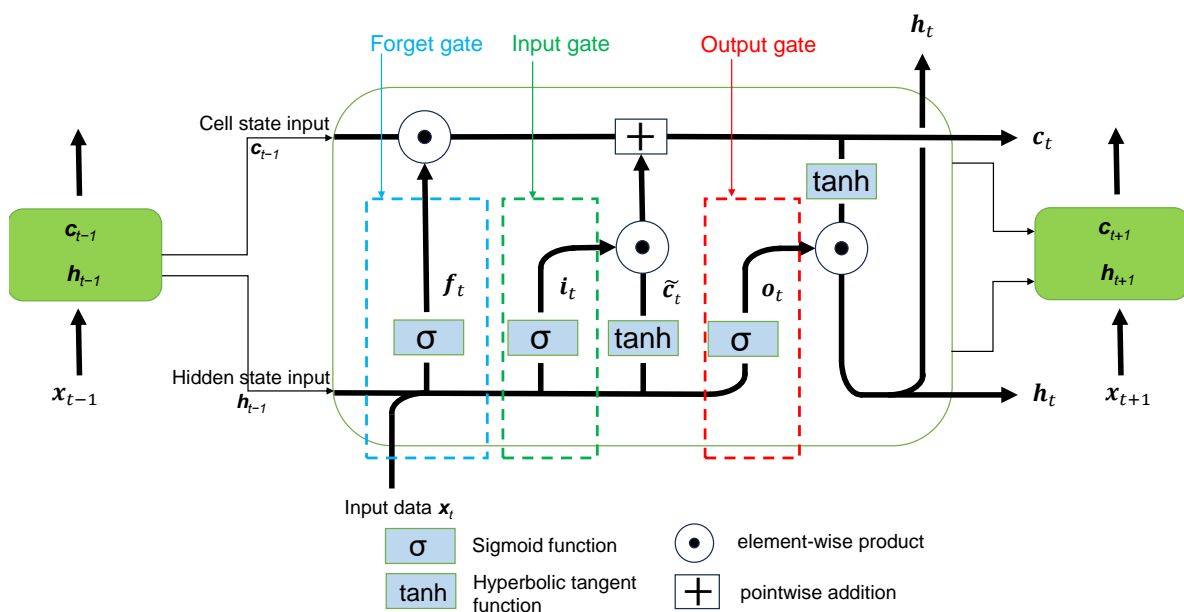


Figure 2. Illustration of the LSTM cell architecture, for the time instances $t - 1, t, t + 1$, with the corresponding gates (Adapted from [68]).

2.4. Transformer-Based Model (Informer)

First off, an initial examination of transformer models is imperative for our investigation. The characteristic feature of transformers is the so-called self-attention mechanism, enabling the model to weigh different parts of the input sequence differently based on their relevance. This attention mechanism allows transformers to capture long-range dependencies efficiently, breaking the sequential constraints of traditional models like LSTMs. The transformer architecture consists of encoder and decoder layers, each comprised of multi-head self-attention mechanisms and feedforward neural

networks. The encoder processes the input sequence using self-attention mechanisms to extract relevant features. The decoder then utilizes these features to generate the output sequence, incorporating both self-attention and encoder-decoder attention mechanisms for effective sequence-to-sequence tasks. The use of positional encoding provide the model with information about the position of input tokens in the sequence. Parallelization becomes a strength, as transformers can process input sequences in parallel rather than sequentially, significantly speeding up training and inference.

The Informer model, which was proposed by Zhou et al. (2021) [52], is a deep artificial neural network, and more specifically an improved Transformer-based model, that aims to solve the Long Sequence Time-series Forecasting (LSTF) problem [52]. This problem requires high prediction capability to capture long-range dependence between inputs and outputs. Unlike LSTM, which relies on sequential processing, transformers, and consequently Informer, leverage attention mechanisms to capture long-range dependencies and parallelize computations. The key feature lies in self-attention, allowing the model to weigh input tokens differently based on their relevance to each other, enhancing its ability to capture temporal dependencies efficiently. The rise in the application of transformer networks have emphasized their potential in increasing the forecasting capability. However, some issues concerning the operation of transformers, such as the quadratic time complexity ($\mathcal{O}(n^2)$), the high memory usage and the limitation of the encoder-decoder architecture, make up transformer to be unproductive in problems when dealing with long sequences. Informer addresses those issues of the "vanilla" transformer [32] and trying to resolve them, by applying three main characteristics [52]:

1. **ProbSparse self-attention mechanism:** a special kind of attention mechanism (ProbSparse - Probabilistic Sparsity) which achieves $\mathcal{O}(L \log L)$ in time complexity and memory usage, where L is the length of the input sequence.
2. **Self-attention distilling:** a technique that highlights the dominant attention making it effective in handling extremely long input sequences.
3. **Generative style decoder:** a special kind of decoder that predicts long-term time series sequences with one forward pass rather than step-by-step, which drastically improves the speed of forecast generation.

Informer model preserves the classic transformer architecture, i.e., encoder-decoder architecture, while at the same time aiming to solve LSTF problem. The LSTF problem, that Informer is trying to solve, has an sequence input at time t :

$$X^t = \{\mathbf{x}_1^t, \mathbf{x}_2^t, \dots, \mathbf{x}_{L_x}^t | \mathbf{x}_i^t \in \mathbb{R}^{d_x}\} \quad (16)$$

and the output or target (predictions) is the corresponding sequence of vectors:

$$Y^t = \{\mathbf{y}_1^t, \mathbf{y}_2^t, \dots, \mathbf{y}_{L_y}^t | \mathbf{y}_i^t \in \mathbb{R}^{d_y}\} \quad (17)$$

where $\mathbf{y}_i^t, \mathbf{y}_i^t$ are input and output series of vectors at a specific time t , and L_x, L_y the input and output number elements, respectively. LSTF encourage a longer enough output length L_y , than usual forecasting cases. Informer supports all forecasting tasks (e.g., multivariate predict multivariate, multivariate predict univariate etc.), where d_x, d_y are feature dimensions of input and output, respectively.

The structure of Informer model is illustrated in Figure 3, wherein three main components are distinguished: encoder layer, decoder layer and prediction layer. Encoder mainly handles longer sequence inputs by using sparse self-attention, an alternative to the traditional self-attention method. The trapezoidal component refers to the extracted self-attention operation, which significantly reduces the network's size. The component between multi-head (i.e., a module for attention mechanisms which is often applied several times in parallel) ProbSparse attention, indicates self-attention distilling blocks (presented with light blue), that lead to concatenated feature map. On the other side, the decoder deals with input from the long-term sequence, padding target elements to zero. This procedure computes an

attention-weighted component within the feature graph, resulting in the prompt generation of these elements in an efficient format.

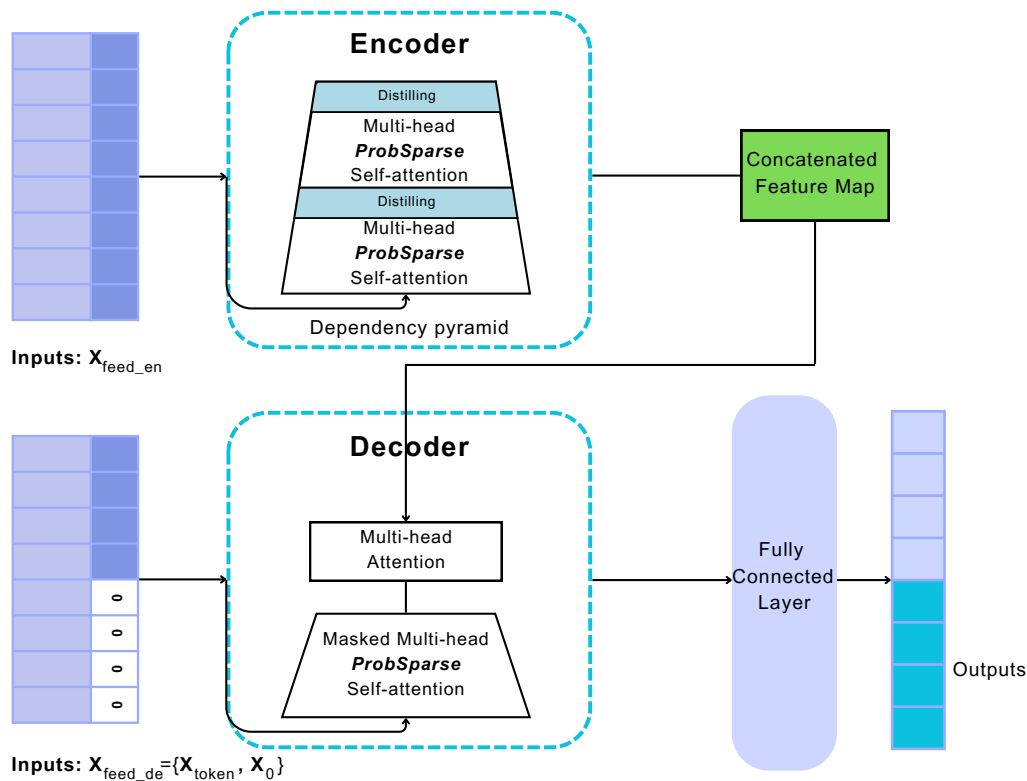


Figure 3. Informer model structure (Adapted from 'Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting', Zhou et al. [52]).

The ProbSparse (i.e., Probabilistic Sparsity) self-attention mechanism reduces the computation cost of self-attention drastically, while maintaining its effectiveness. Essentially, instead of Equation (18), Informer uses Equation (19) for the calculation of attention. The core concept of the attention mechanism revolves around computing attention weights for each element in the input sequence, indicating their relevance to the current output. This process begins by generating query (q), key (k), and value (v) vectors for each element in the sequence, where the query vector represents the current network output state, and the key and value vectors correspond to the input element and its associated attribute vector, respectively. Subsequently, these vectors are utilized to calculate attention weights through a scaled dot-product relation, measuring the similarity between the vectors. Figure 4 provides an overview of the attention mechanism's architecture, where MatMul stands for matrix multiplication, SoftMax is the normalized exponential function for converting numbers into a probability distribution, and Concat for concatenation. The function that summarises attention mechanism can be written as [32]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \quad (18)$$

$$\text{Attention}(Q, K, V)_{\text{Informer}} = \text{softmax}\left(\frac{\bar{Q}K^T}{\sqrt{d}}\right)V \quad (19)$$

where Q, K, V are the input matrices of the attention mechanism. The dot-product of query q and key k is divided by the scaling factor of \sqrt{d} , with d representing the dimension of query and key vectors (input dimension). The significant difference between Equation (18) and 19 is that keys K are not

linked to all possible Q queries but to the u most dominant ones, resulting in the matrix \bar{Q} (which is obtained by the probability sparse of Q).

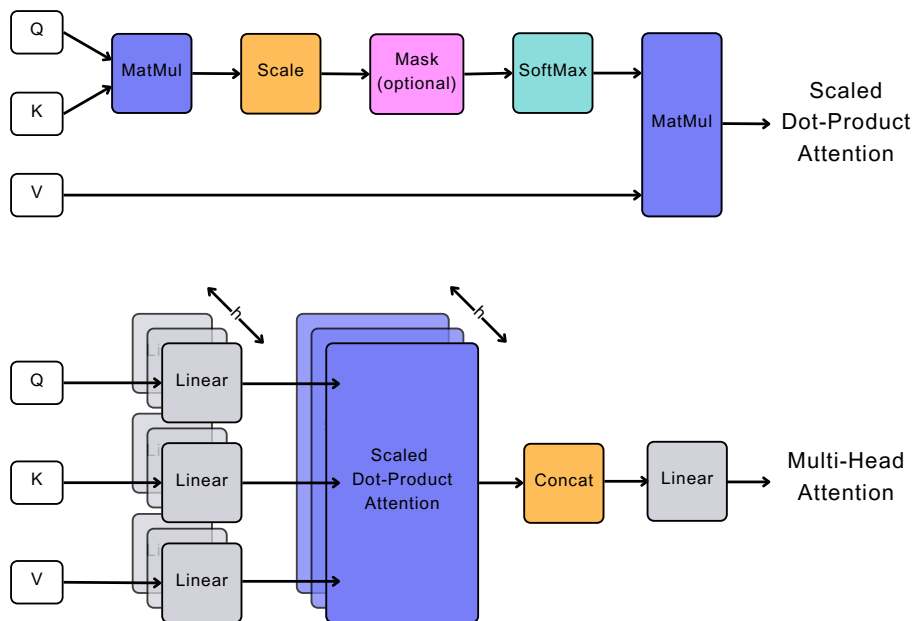


Figure 4. Visualization of attention mechanism (Adapted from ‘Attention Is All You Need’, Vaswani et al. [32]).

An empirical approach is proposed for effectively measuring query sparsity, where is calculated by a method similar to Kullback–Leibler divergence. For each (query) $q_i \in R^d$ and $k_j \in R^d$ in the set of keys K , there is the bound $\ln L_K \leq M(q_i, K) \leq \max_j \{q_i k_j^T / \sqrt{d}\} - \frac{1}{L_K} \sum_{j=1}^{L_K} q_i k_j^T / \sqrt{d} + \ln L_K$ [52]. Therefore, the proposed max-mean measurement is as follows:

$$\bar{M}(q_i, K) = \max_j \left\{ \frac{q_i k_j^T}{\sqrt{d}} \right\} - \frac{1}{L_K} \sum_{j=1}^{L_K} \frac{q_i k_j^T}{\sqrt{d}} \quad (20)$$

If the i , h query achieves a higher $\bar{M}(q_i, K)$, its attention probability p becomes more "diverse" and is likely to include the dominant dot-product pairs in the header field of the long-tail self-attention distribution.

The distribution of self-attention scores follows a long-tail distribution, with "active queries" positioned at the upper end of the feature map within the "head" scores, while "lazy queries" reside at the lower end within the "tail" scores. The preceding is illustrated in Figure 5. The ProbSparse mechanism is tailored to prioritize the selection of "active" queries over "lazy" ones.

The encoder aims to capture the long-range dependencies within the lengthy sequence of inputs, under the constraint of using memory. Figure 6 is depicting model’s encoder. The process of self-attention distilling serves to assess the dominant features and generate a consolidated map of self-attention features for the subsequent level. Essentially, it involves simplifying the complex self-attention mechanism of the transformer model into simpler and smaller form, suitable for integration into the Informer model.

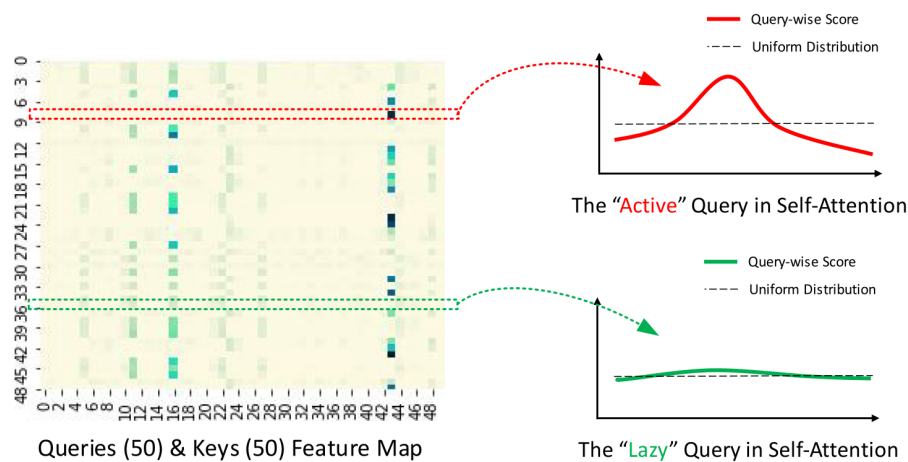


Figure 5. Illustration of ProbSparse Attention (Source: [69]).

As distinguished in Figure 6 encoder contains enough attention blocks, convolution layers (Conv1d) and max pooling layers (MaxPool), to encode the input data. The copies of the main stack (Attention Block 1) with continuously decreasing inputs by half, increase the reliability of the distilling operation. The process continues by reducing to 1/4 of the original length. At the end of the encoder, all feature maps are concatenated to direct the output of the encoder directly to the decoder.

The purpose of the 'distillation' operation is to reduce the size of the network parameters, giving higher weights to the dominant features, and produce a focused feature map in the subsequent layer. The distillation process from the j -th layer to the $(j + 1)$ -th layer is described by the following procedure:

$$X_{j+1}^t = \text{MaxPool}(\text{ELU}(\text{Conv1d}([X_j^t]_{\text{AB}}))) \quad (21)$$

where $[X_j^t]_{\text{AB}}$ represents the attention block, $\text{Conv1d}(\cdot)$ performs an 1-D convolutional filters with the $\text{ELU}(\cdot)$ activation function [70].

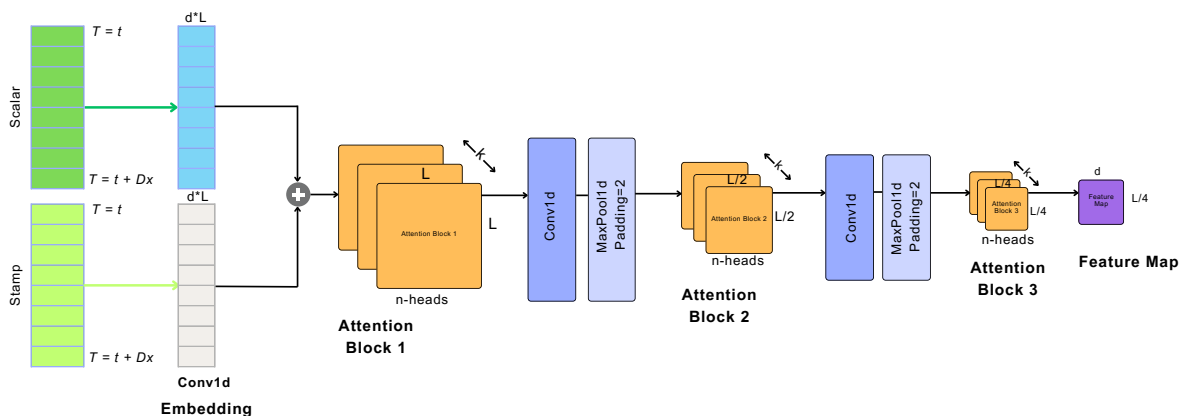


Figure 6. Informer's encoder architecture (Adapted from [52]).

The structure of the decoder does not differ greatly from that published by Vaswani et al. [32]. It is capable of generating large sequential outputs through a single forward pass process. As shown in Figure 3, it includes two identical multi-head attention layers. The main difference is the way in which the predictions are generated through a process called generative inference, which significantly speeds up the long-range prediction process. The decoder receives the following vectors:

$$\mathbf{X}_{\text{de}}^t = \text{Concat}(\mathbf{X}_{\text{token}}^t, \mathbf{X}_0^t) \in \mathbb{R}^{(L_{\text{token}} + L_y) \times d_{\text{model}}} \quad (22)$$

where $\mathbf{X}_{\text{token}}^t \in \mathbb{R}^{L_{\text{token}} \times d_{\text{model}}}$ is the start token (i.e., parts of input data), $\mathbf{X}_0^t \in \mathbb{R}^{L_y \times d_{\text{model}}}$ is a placeholder for the target sequence (set to 0). Subsequently, the masked multi-head attention mechanism adjusts the ProbSparse self-attention computation by setting the inner products to $-\infty$, thereby prohibiting each position from attending to subsequent positions. Finally, a fully connected layer produces the final output, while the dimension of the output d_y depends on whether univariate or multivariate prediction is implemented.

2.5. Evaluation Metrics

To assess the effectiveness of the different models accurately, this paper engages Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE) and Mean Absolute Percentage Error (MAPE) as evaluation metrics. In some cases, the coefficient of determination R^2 was also determined. The formulas for calculating these metrics are provided below.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2 \quad (23)$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2} \quad (24)$$

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i| \quad (25)$$

$$\text{MAPE} = \frac{1}{n} \sum_{i=1}^n \frac{|y_i - \hat{y}_i|}{y_i} \cdot 100\% \quad (26)$$

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2} \quad (27)$$

where y_i is the i -th observed value; \hat{y}_i is the corresponding forecasted value; \bar{y} is the mean of the y values; and n the number of total data points for y_i .

Materials and Methods should be described with sufficient details to allow others to replicate and build on published results. Please note that publication of your manuscript implicates that you must make all materials, data, computer code, and protocols associated with the publication available to readers. Please disclose at the submission stage any restrictions on the availability of materials or information. New methods and protocols should be described in detail while well-established methods can be briefly described and appropriately cited.

Research manuscripts reporting large datasets that are deposited in a publicly available database should specify where the data have been deposited and provide the relevant accession numbers. If the accession numbers have not yet been obtained at the time of submission, please state that they will be provided during review. They must be provided prior to publication.

Interventionary studies involving animals or humans, and other studies require ethical approval must list the authority that provided approval and the corresponding ethical approval code.

This is an example of a quote.

3. Analysis and Results (Case Study 1: River Test)

3.1. Study Area

The study area centers on the River Test [71] (Figure 7), situated in Hampshire, southern England. It rises at Ashe close to Basingstoke, it spans approximately 64 km², flowing southwards until it

meets Southampton Water. The River Test is an important watercourse known for its ecological diversity, historical and hydrological significance. Notably, it encompasses a significant biological area designated as a Special Scientific Interest, spanning 438 hectares (ha).

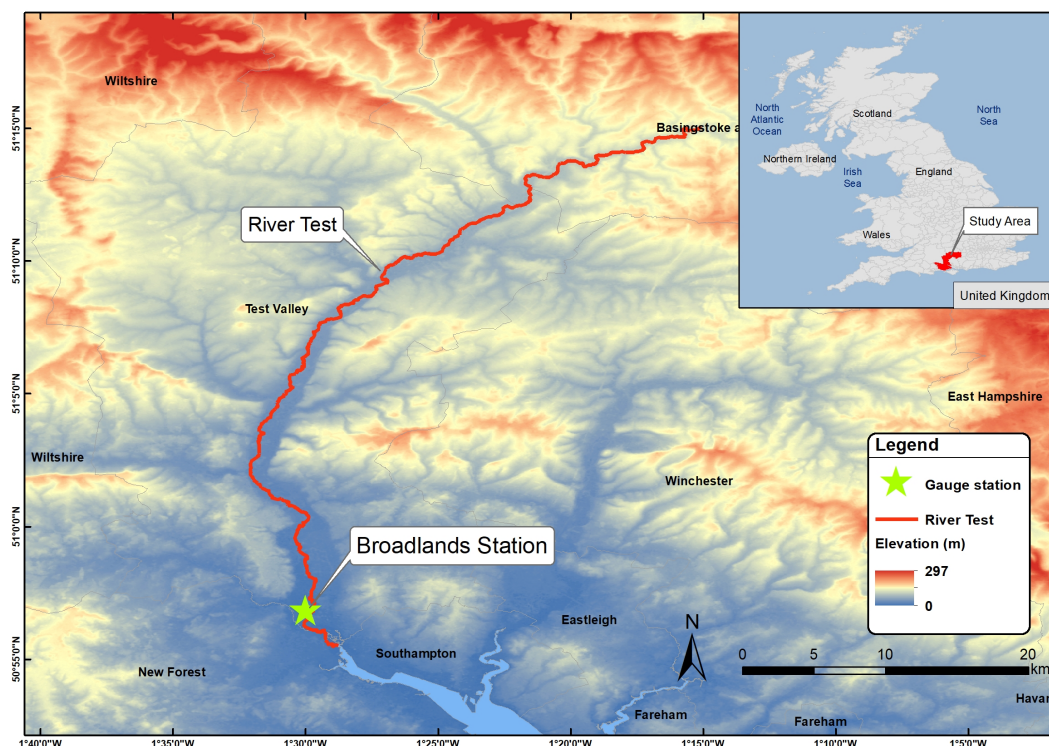


Figure 7. Study area.

3.2. Data

The hydroclimatic data were collected, stored, processed and consolidated, by national data centres in England. In particular, data from The CEDA Archive [72], Defra [73], and the National River Flow Archive [74] were utilized. Dataset contains 12 features, among which flow from river stations and climate data of the region, i.e., precipitation and temperature across diverse points within the region of Hampshire in England. The dataset comprises 41 years of daily observations spanning from 1980 to 2021, totaling 15 341 values (slightly fewer in certain attributes). Figure 8 presents the first 20 rows of the data in tabular format, displaying all their attributes, while Table 2 provides a description of these characteristics. Table 1 shows the metadata information about the stations of the dataset.

Table 1. Metadata of stations of the dataset.

Station name	Type of variable	Longitude	Latitude
Marborough	rain	-1.73	51.41
Andover	rain	-1.48	51.2
Andover	temp	-1.47	51.22
Bath	rain	-2.36	51.38
Ower	rain	-1.53	50.95
Quidhampton	rain	-1.26	51.25
Salisbury	rain	-1.79	51.07
Stockbridge	rain	-1.56	51.10
Ower	flow	-1.53	50.95
Broadlands	flow	-1.50	50.95

	date	rainfall_Marborough	rainfall_Andover	rainfall_Bath	rainfall_Ower	rainfall_Quidhampton	rainfall_Salisbury	rainfall_Stockbridge	tasmax_Andover	tasmin_Andover	flow_Broadlands	flow_Ower
0	1/1/1980	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.44	-3.86	13.45	1.03
1	1/2/1980	0.85	0.25	1.16	0.36	1.05	0.68	0.59	3.26	-5.86	13.01	0.90
2	1/3/1980	12.03	11.89	5.86	10.27	11.27	10.04	12.34	10.49	-5.42	13.48	2.42
3	1/4/1980	0.92	0.19	1.18	0.47	0.28	0.54	0.30	8.69	2.71	16.15	2.73
4	1/5/1980	0.08	0.13	0.22	0.01	0.01	0.02	0.02	7.95	1.30	14.43	1.32
5	1/6/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	7.56	2.51	13.63	1.08
6	1/7/1980	0.05	0.01	0.02	0.00	0.01	0.00	0.00	5.16	0.32	13.09	1.07
7	1/8/1980	0.00	0.00	0.00	0.00	0.01	0.00	0.00	4.74	0.53	12.69	1.06
8	1/9/1980	0.00	0.00	0.01	0.00	0.00	0.02	0.00	3.06	-1.52	12.52	0.82
9	1/10/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	2.79	-0.44	12.31	0.55
10	1/11/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.52	-3.44	11.98	0.51
11	1/12/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.22	-3.93	11.88	0.48
12	1/13/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.30	-4.57	11.80	0.47
13	1/14/1980	3.73	0.54	5.81	0.02	0.42	2.00	0.27	0.56	-5.13	11.71	0.47
14	1/15/1980	0.18	0.28	0.16	0.00	0.38	0.10	0.03	2.07	-4.36	11.24	0.46
15	1/16/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.55	-0.41	11.61	0.46
16	1/17/1980	0.00	0.00	0.00	0.00	0.00	0.00	0.00	3.60	-1.70	11.58	0.48
17	1/18/1980	0.05	0.11	0.00	0.01	0.03	0.36	0.46	1.35	-1.73	11.49	0.55
18	1/19/1980	2.97	2.11	1.23	4.29	2.11	4.27	3.12	4.23	-1.99	11.41	0.67
19	1/20/1980	15.12	17.80	11.23	17.23	13.77	18.35	16.97	6.74	-0.26	11.53	1.12

Figure 8. River Test dataset format.

Table 2. Description of data features.

Features	Description	Units
date	measurement date	d
rainfall_{location}	the observed daily rainfall at the location	mm
tasmax	the maximum daily observed temperature	°C
tasmin	the minimum daily observed temperature	°C
flow_{location}	the daily flow at the specific location	m ³ /s

During the data preprocessing stage, we conducted checks to identify any erroneous values, temporal gaps or potential outliers in the data. In this study we focus on streamflow forecasting, while the `flow_Broadlands` column was selected for the method application. This column represents the measured flow in the Broadlands area (Figure 7). The fluctuation in flow is closely associated with flooding events, rendering this data characteristic particularly important.

The streamflow measurements consists of 15 225 values from 1/1/1980 to 6/9/2021. We split the data into training, validation, and testing sets using a ratio of 0.7:0.1:0.2, a fairly common splitting scheme. Approximately, the initial 33 years were used for model training, with the remaining data allocated for testing. Figure 9 illustrates the time series of the station streamflow, with the training (train), validation (val) and testing (test) data split indicated.

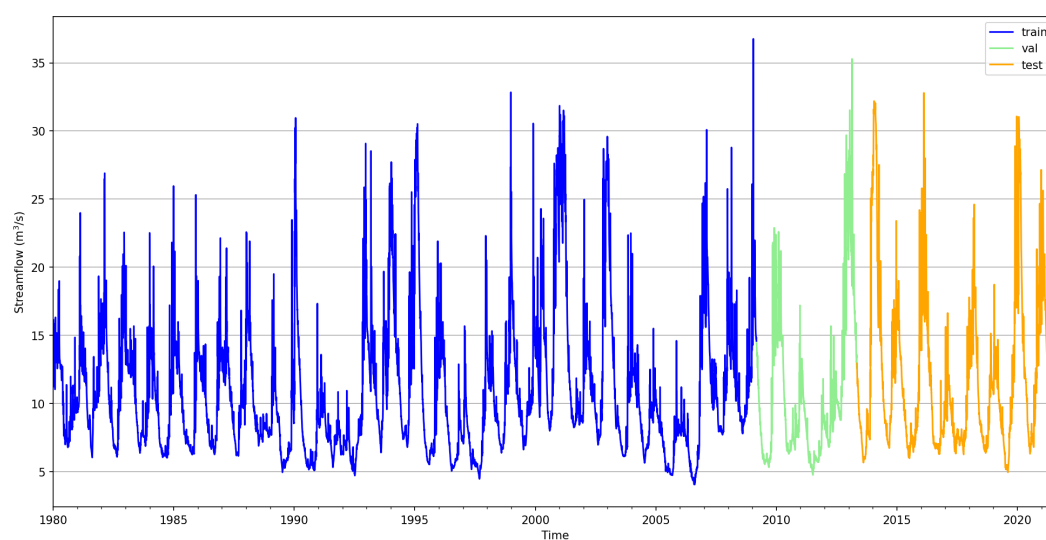


Figure 9. River streamflow time series of Broadlands station.

In this particular case study we aim to forecast streamflow time series at the Broadlands station using various forecast horizons. The selected forecast horizons include 2, 10, 20, 40, 60, 80, 100, and 168 days. Four different models were employed: a common time series model (ARIMA), a Hurst-Kolmogorov stochastic model (SB), and two deep learning networks (LSTM and Informer). All analyses and code development were conducted in Python, utilizing relevant libraries. The computational resources of Google Colab, specifically the NVIDIA Tesla K80 with 12 GiB VRAM, were leveraged to ensure optimal conditions for training deep neural networks.

3.3. Model architectures

3.3.1. SB

For this application, the stochastic benchmark model, described in subSection 2.1, was chosen over a naive model. The naive approach assumes that future values in a time series will mirror the most recent observed value, or the mean of all value of time series. In contrast, the SB model determines the precise number of past terms to consider and computes a representative average for the forecast horizon at hand. This model serves as a benchmark for comparison against more sophisticated forecasting models.

In order to find the required conditions (in our case days) to be taken into account regarding the forecast horizon, it is necessary to calculate the Hurst coefficient H , which is approximated by the climacogram. Figure 10 shows the climacogram of the time series of streamflow at Broadlands station. For simplicity, only the long-term dependence was considered (highlighted with orange in Figure 10), rather than the entire climacogram. This resulted in a coefficient $H = 0.7$, which simplifies things and equations of Section 2.1 are directly applicable. Utilizing Equation (8) we compute the required ν based on given κ and H . Table 3 summarizes those combinations of ν , κ .

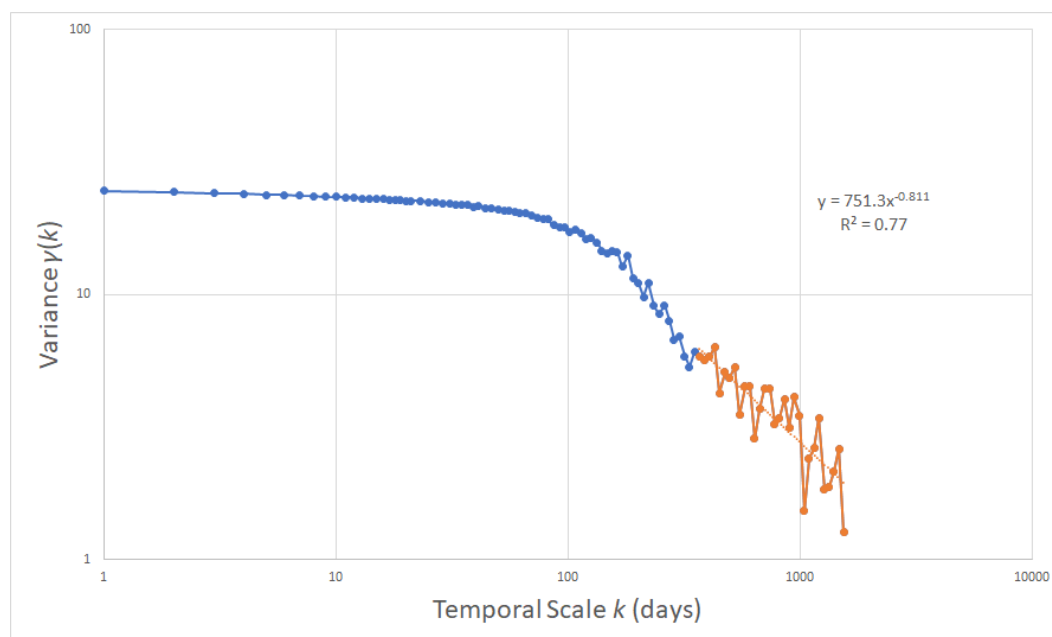


Figure 10. Empirical climacogram of streamflow at Broadlands station.

Table 3. Ratio of terms (in days) of past ν and future κ .

κ (days)	2	10	20	40	60	80	100	168
ν (days)	64	320	640	1280	1920	2560	3200	5376

3.3.2. ARIMA

By monitoring the AIC metric values across different parameter combinations, as well as their corresponding time, it was determined that the optimal parameters for the ARIMA model are ARIMA(3,1,3). For this specific model the AIC was found to be $AIC = 34\,941.93$ with a corresponding computational time of 9.7 seconds.

3.3.3. LSTM

The LSTM model was constructed using the PyTorch library, a widely used tool for deep learning tasks. During training, the time series data was utilized, and fixed-length sequences were generated after normalization. Sequentially, each sequence was fed into the LSTM model individually, with the immediate subsequent value also recorded. Through testing and experimentation with different architectures, the optimal hyperparameters and network architecture were identified, as summarized in Table 4. This iterative process involved adjusting and refining the model to achieve the best performance.

Table 4. LSTM model Hyperparameters.

Hyperparameters	Values
Neurons	100
Dense Layers (Fully Connected)	5
Dropout	0.2
Batch size	16
Sequence length	180
Optimizer	Adam
Activation function	ReLU
Loss function	mse
Learning rate	$5 \cdot 10^{-4}$
Epochs	20

3.3.4. Informer

The final and main model examined in this study is the transformer-based Informer model. Following data partitioning into training, validation, and test sets, normalization is applied, and the data is segmented into input and target subsequences. The input subsequences are then passed through the encoder to the decoder, where future values of the time series are predicted. Table 5 outlines the model architecture, excluding details on the encoder input sequence length and prediction sequence length. Through an iterative fine-tuning process involving various combinations of input and prediction sequences, optimal values, as presented in Table 6, were determined.

Table 5. Informer model Hyperparameters.

Hyperparameters	Values
Encoder input size	2
Decoder input size	2
Dimension of model	512
Number of heads	8
Encoder layers	512
Decoder layers	512
Dropout	0.05
Attention	ProbSparse
Activation function	GELU
Batch size	32
Optimizer	Adam
Loss function	mse
Learning rate	10^{-4}
Epochs	8
Early stopping patience	3

Table 6. Combinations of input sequence length and prediction length of Informer model.

prediction length (days)	2	10	20	40	60	80	100	168
sequence length (days)	200	100	150	300	300	300	300	250

3.4. Results and comparison

As previously mentioned, the forecast horizon spans from 2 to 168 days. Table 7 provides a detailed overview of the performance metrics, including MSE, RMSE, MAE, and MAPE, for each of the forecast horizons, across all models. For each forecast horizon, the best scores are emphasized in bold. Figure 11 provides a comprehensive and straightforward representation of the data presented in the corresponding table, through a bar chart depicting evaluation metrics. Furthermore, Figure 12 depict the predicted and ground truth values in corresponding prediction window.

Insights can be derived from the findings presented in Table 7. The deep learning models exhibit superior performance compared to both the naive stochastic approach and the statistical ARIMA model, which struggles to fully grasp the complex dynamics of time series. Specifically, LSTM demonstrates slightly enhanced outcomes at the 10-day forecasting horizon, whereas Informer model emerges as the top performer overall. Remarkably, the Informer model substantially enhances the results, with the prediction error gradually and smoothly increasing as the prediction horizon extends. This underscores the Informer's effectiveness in improving prediction capabilities for long-range forecasting tasks. The Informer's performance, as measured by the MAPE metric, does not surpass 10%, demonstrating its high level of accuracy.

Table 7. Models results of streamflow forecasting on River Test.

Method	Metric	2	10	20	40	60	80	100	168
SB	MSE	14.302	22.519	0.748	1.627	3.091	5.252	6.314	14.798
	RMSE	3.782	4.745	0.865	1.276	1.758	2.292	2.513	3.847
	MAE	3.775	4.721	0.767	1.062	1.527	1.994	2.179	3.459
	MAPE	0.297	0.397	0.068	0.107	0.168	0.240	0.296	0.502
ARIMA	MSE	0.527	2.224	3.356	9.092	14.197	19.196	25.933	32.290
	RMSE	0.726	1.491	1.832	3.015	3.768	4.381	5.092	5.682
	MAE	0.689	1.417	1.756	2.781	3.499	4.082	4.718	5.376
	MAPE	0.055	0.120	0.154	0.279	0.381	0.481	0.625	0.758
LSTM	MSE	0.362	0.105	0.656	0.407	0.304	0.349	1.047	5.520
	RMSE	0.601	0.325	0.810	0.638	0.551	0.591	1.023	2.349
	MAE	0.589	0.261	0.637	0.482	0.413	0.452	0.746	1.802
	MAPE	0.046	0.021	0.056	0.045	0.041	0.050	0.104	0.255
Informer	MSE	0.006	0.311	0.094	0.091	0.269	0.124	0.760	0.648
	RMSE	0.077	0.558	0.307	0.301	0.519	0.352	0.872	0.805
	MAE	0.069	0.502	0.251	0.241	0.465	0.278	0.720	0.564
	MAPE	0.005	0.043	0.022	0.024	0.049	0.032	0.095	0.083

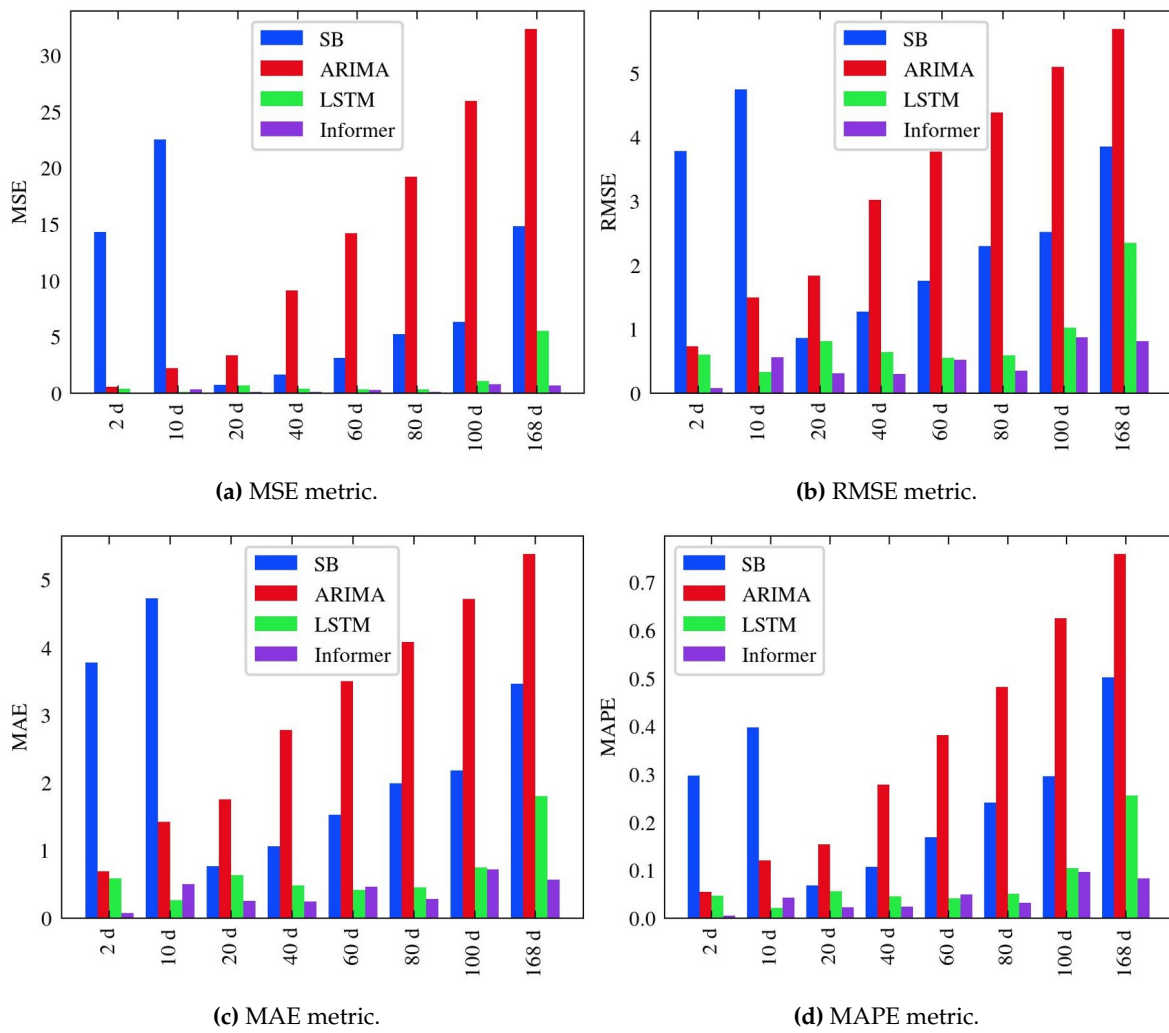
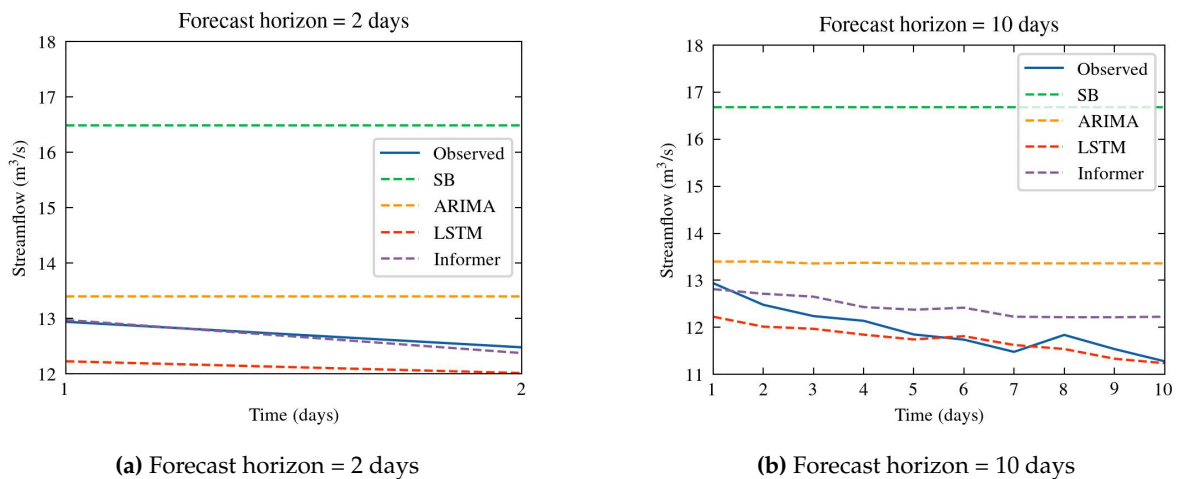


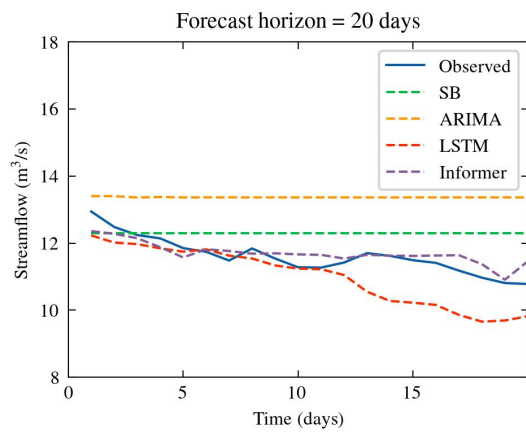
Figure 11. Bar chart presenting the evaluation metric results.



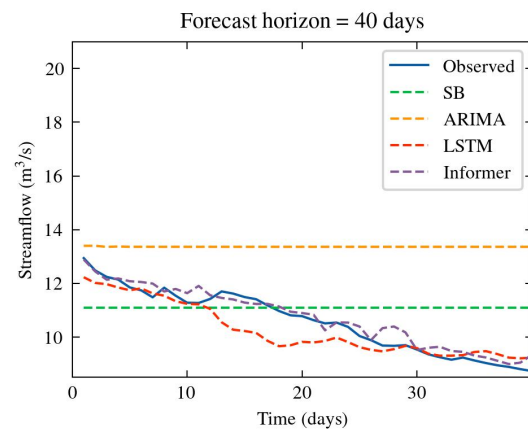
(a) Forecast horizon = 2 days

(b) Forecast horizon = 10 days

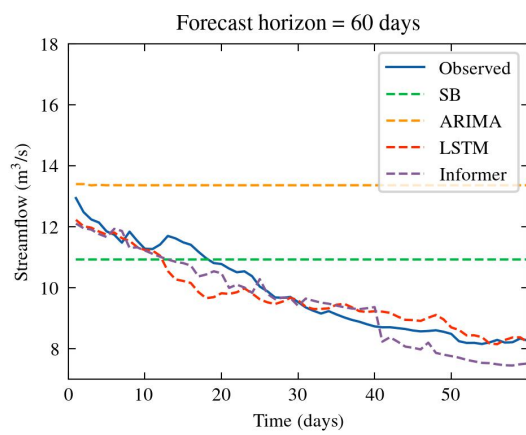
Figure 12. Cont.



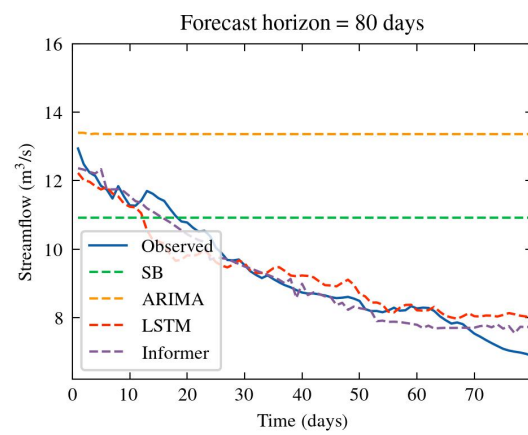
(c) Forecast horizon = 20 days



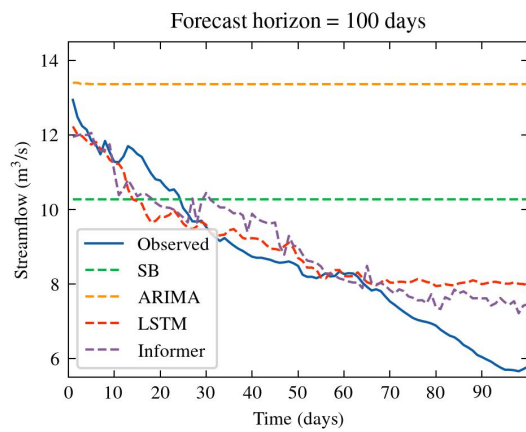
(d) Forecast horizon = 40 days



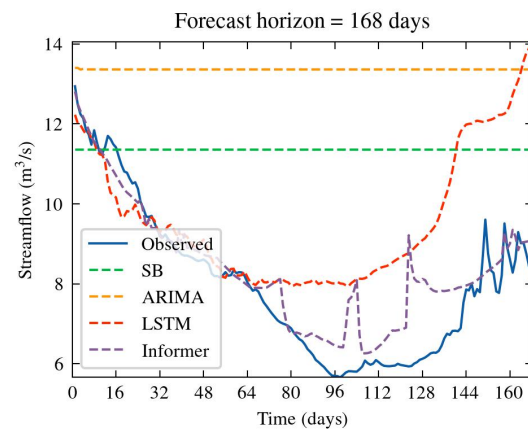
(e) Forecast horizon = 60 days



(f) Forecast horizon = 80 days



(g) Forecast horizon = 100 days



(h) Forecast horizon = 168 days

Figure 12. Prediction results of four models compared with ground truth across all prediction windows.

In the model comparison, the Informer model stand out, having the best performance. Figure 13 presents the Informer model's outcomes in a scatter plot displaying predicted streamflow values. The solid blue line in the plot represents the ideal estimation, where if each prediction point of the model matched the actual observed value. Overall, Informer model demonstrates good performance, with the trend line exhibiting a coefficient of determination (R^2) of 0.91. However, weaker performance is noted at lower values, corresponding to local minima in our context.

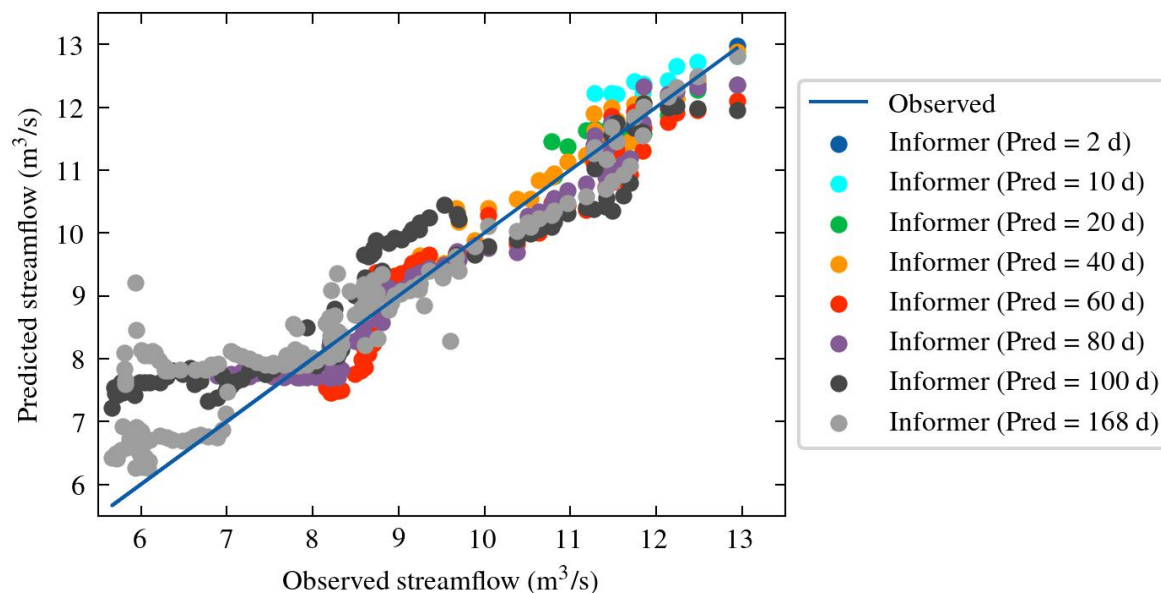


Figure 13. Scatter plot of predicted streamflow values from the Informer model in the River Test.

4. Analysis and Results (Case Study 2: Examining the performance of the Informer model across numerous rivers)

4.1. Study Area

The selected study area for evaluating the Informer performance across a substantial number of time series (>100) covers a wider area of the United Kingdom, where the previous case study belongs as well.

Given its varied topography and extensive river networks, the UK frequently witnesses devastating flooding incidents with far-reaching consequences. This study centers on the rivers of the UK, aiming to devise and assess a deep learning-based method for forecasting flood events, and more specifically the Informer model.

A key factor motivating the selection of this study area is the extensive database maintained by the United Kingdom, known as the National River Flow Archive (NRFA) [74], which systematically gathers and manages data on river flow from numerous gauging stations across the entire country. This database serves as a fundamental resource for the current investigation. Figure 14 depicts the extensive coverage of gauging stations within the NRFA database, along with the corresponding river networks.



Figure 14. Map of UK with the locations of available stations from the NRFA database (Source: <https://nrfa.ceh.ac.uk/data/search> (accessed on 10 February 2023)).

4.2. Data

As already mentioned, the data were obtained from the NRFA database [74] and are available online. Similar to the prior study, this analysis focuses on time series data of river streamflow, aiming to forecast for a specific horizon. Accessing this database allows for straightforward searches, leveraging filters based on specific criteria relevant to individual research needs (e.g., real-time data availability, location etc.). In this particular study, the selection was primarily based on three criteria:

1. The availability of daily streamflow data.
2. The time series should span a long period (i.e., > 20 years)
3. The data should be continuous without missing values, with a completeness rate exceeding 95%.

This analysis resulted a collection of 112 data stations, with the their respective geographical locations illustrated in Figure 15. The dataset collected represents a broad spectrum of river morphologies, hydroclimate conditions, and varying amounts of historical data.

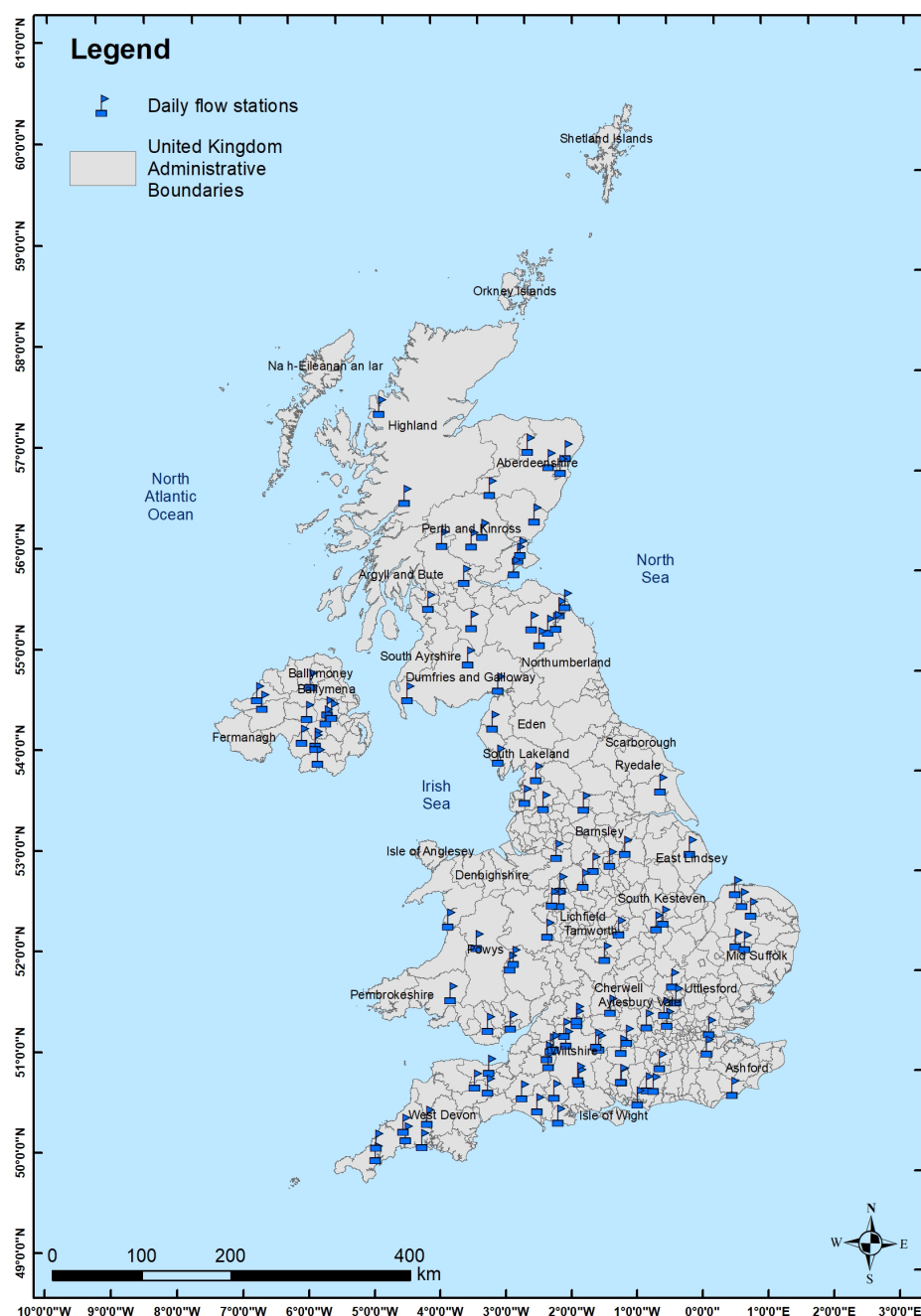


Figure 15. Geographical distribution of the 112 stations used for forecasting.

4.3. Informer Architecture

The evaluation using the Informer model was conducted uniformly across all time series with identical architecture and forecast horizon (30 days). Since the study's aim wasn't to fine-tune forecast results for each individual river, conducting a thorough hyperparameters search on a river-by-river basis was considered unnecessary, computationally expensive and time-consuming. Based on the case study presented in Section 3, we identified the optimal architecture tailored to the specified forecast horizon. Thus, the selection of input and output sequences mirrored that of the previous River Test study and remained consistent. Following data split into training, validation, and testing sets (maintaining a 70:10:20 ratio), normalization and segmentation into input and target subsequences were performed as previously. Table 8 outlines the details on the model architecture employed.

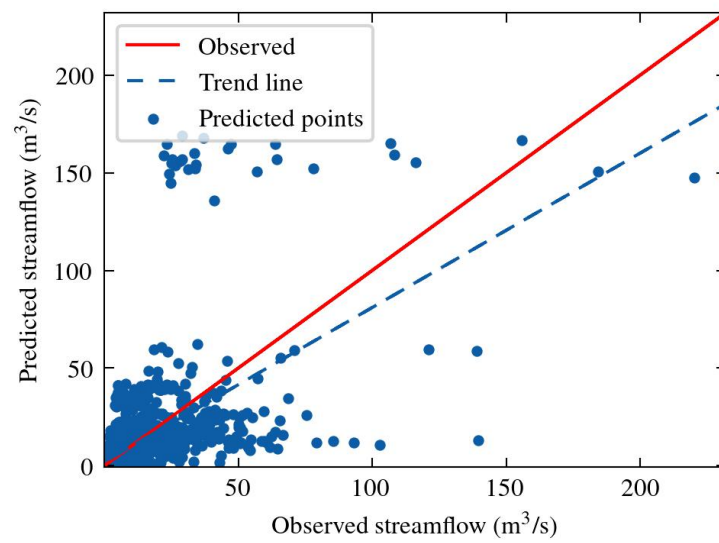
Table 8. Informer model Hyperparameters.

Hyperparameters	Values
sequence length of encoder	300
label length of decoder	48
prediction length	30
Encoder input size	2
Decoder input size	2
Dimension of model	512
Number of heads	8
Encoder layers	512
Decoder layers	512
Dropout	0.05
Attention	ProbSparse
Activation function	GELU
Batch size	32
Optimizer	Adam
Loss function	mse
Learning rate	10^{-4}
Epochs	8
Early stopping patience	3

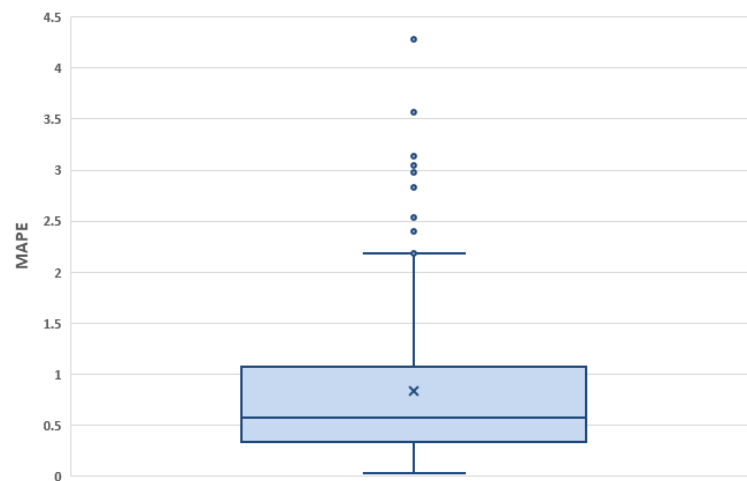
4.4. Results

For the 30-day forecast horizon and utilizing the same architecture, the Informer model was trained to forecast the river flow for each station. It's important to highlight that this approach isn't ideal for deep learning-based forecasting tasks, as each time series should ideally be individually treated and optimized. However, addressing such a significant quantity of time series in this manner demands significant computational resources and time. Hence, for this specific scenario, the investigation focused on evaluating the model's performance using a fixed network architecture.

Figure 16 summarises the prediction results of all 112 stations in a single scatter plot between observed and estimated flows. The figure also provides the optimal prediction (red solid line), where the forecast coincides with the ground truth. However, the trend line of the predictions (blue dashed line) exhibits a coefficient of determination $R^2 = 0.4$, implying poor model performance. Figure 16 also shows the box plot illustrating the MAPE performance metrics. Specifically, for every 30-day forecast of each time series, the MAPE metric was computed once. The sum of these errors, for each time series, is plotted in the box plot. The diagram depicts the moderate performance of the Informer network, characterized by a median value of 0.5 or 50%.



(a)



(b)

Figure 16. (a) Scatter plot of predicted Informer model values and (b) box plot of the MAPE error metric between estimated and observed streamflow, examined at 112 stations.

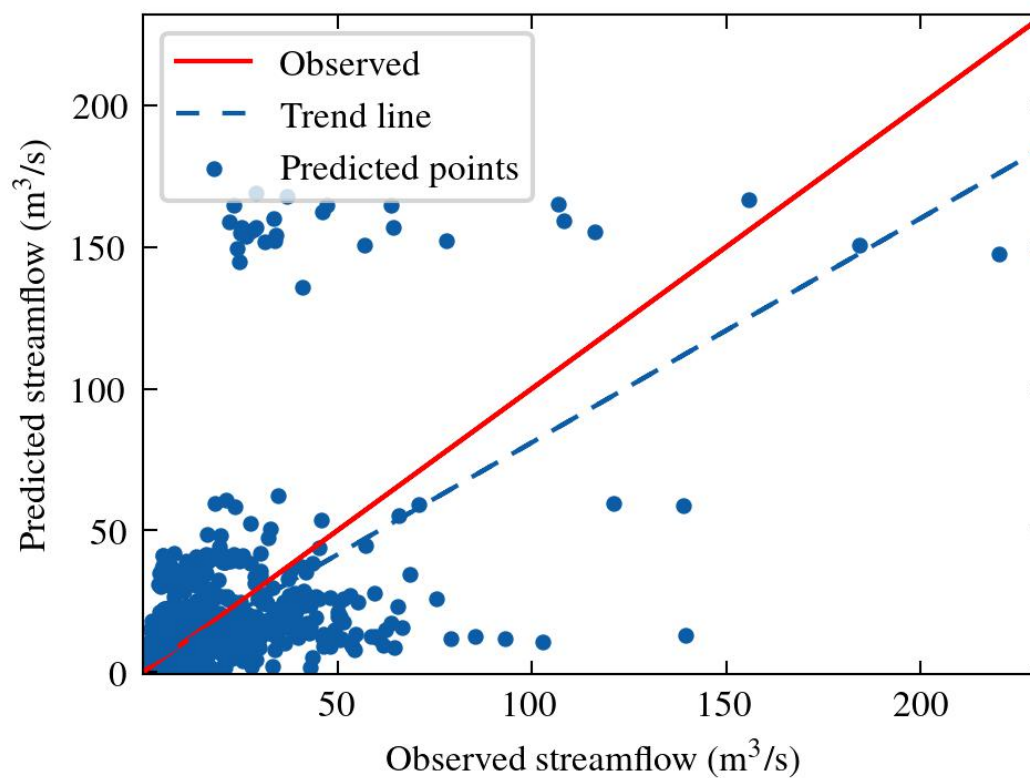


Figure 17. Scatter plot of predicted Informer model values, examined at 112 stations.

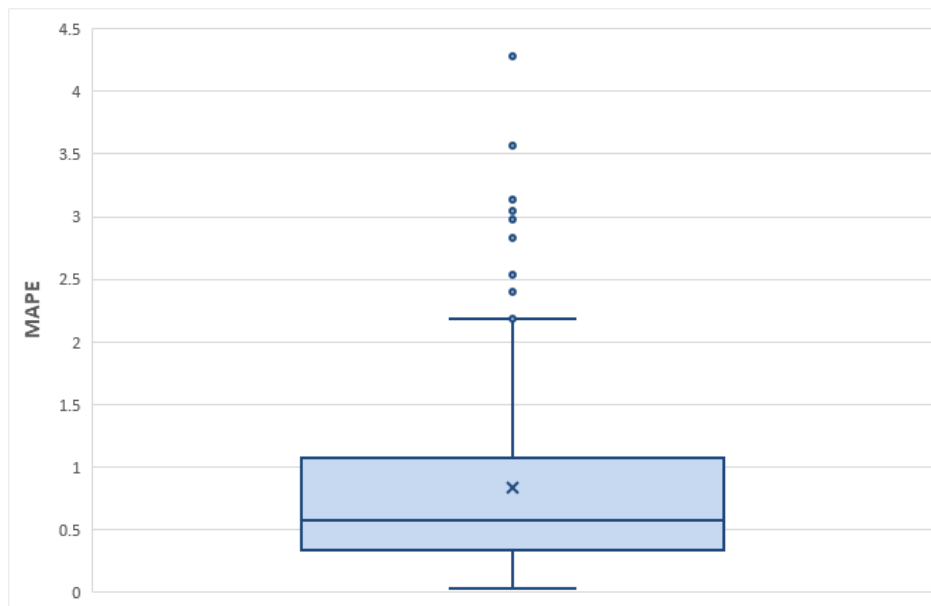


Figure 18. Box plot of the MAPE error metric between estimated and observed streamflow.

To examine the underperformance of the model, we analysed the scores of loss functions across the three subdivisions of all time series (i.e., train, validation and test set), relative to the structure of the time series, as represented by climacograms. Figure 19 displays all the empirical climacograms. The entire 112 climacograms were sorted according to their slope, and subsequently, their respective losses on the train, validation, and test sets were assigned to each one. The results of the earlier investigation are plotted in Figure 20.

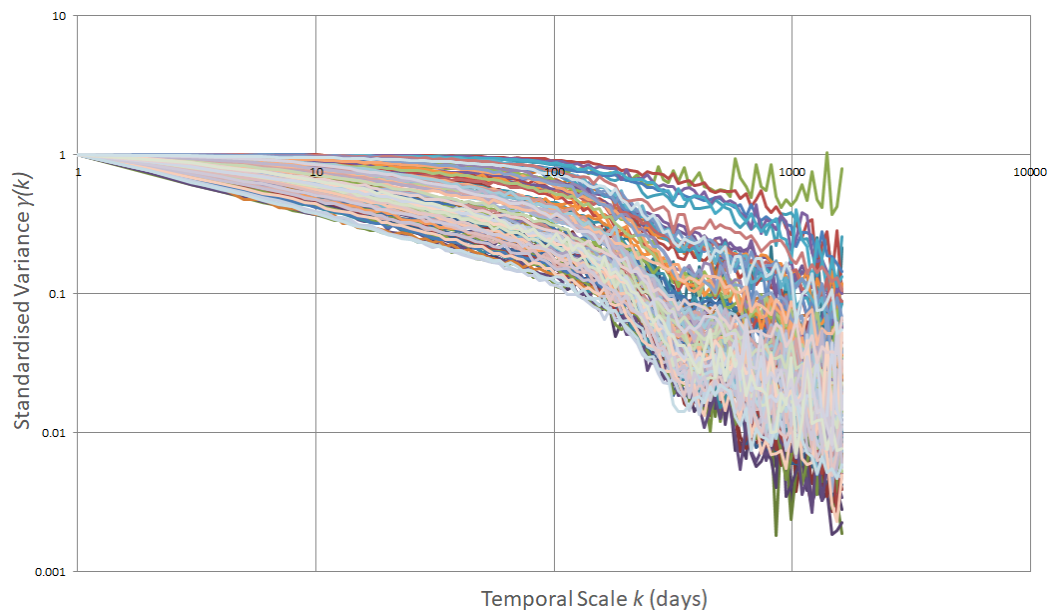


Figure 19. Empirical climacograms of the 112 daily streamflow time series used in the analysis.

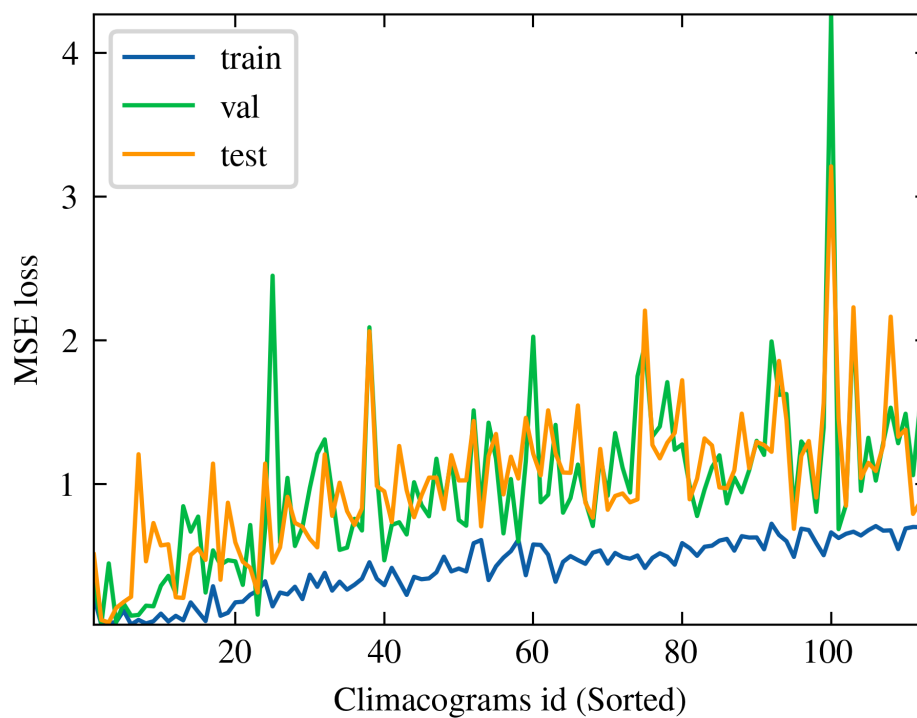


Figure 20. Informer performance compared with climacograms. On horizontal line each climacogram is sorted based on slope, in ascending order. Climacogram with id near to zero indicates higher H value, which decreases as we move towards the right of the axis.

5. Discussion and Future directions

In the past few years, transformer network and the variations of its architecture have been extensively applied across many domains [33]. Although the high popularity they receive, their usage in hydrology and streamflow forecasting is still in its early stages and remains relatively underdeveloped. In the above sections we tried to make some insights about streamflow forecasting with a transformer-based model. i.e. Informer. We utilized real-world data to assess the performance

of the models, evaluating results with widely used metrics—MSE, RMSE, MAE, MAPE and R^2 . When some studies analyse flood prediction with transformer models, they typically rely on data from only one or two stations to generate their results (e.g., [51,56,75]). In contrast, we also make a case study where we focus on Informer model and its performance on >100 river streamflow stations.

In the initial application, where data from River Test were analysed, we compared the Informer model, the LSTM model, the ARIMA model and the Hurst-Kolmogorov stochastic approach. Data include daily streamflow of Broadlands station and the chosen forecast horizons were 2, 10, 20, 40, 60, 80, 100, and 168 days. Deep learning approaches have the best results with the Informer showing its superiority over LSTM—especially for long-term prediction (Figure 12). It should be noted that SB and ARIMA models could have better results if the seasonality was explicitly modeled. Stochastic approach could replace the classic baseline models in hydrology and especially in forecasting problems, as it is theoretically authenticated. Based on our results (Figure 13) Informer model has pretty good performance with a coefficient of $R^2 = 0.91$. The optimization task in transformer networks is a pretty challenging work, due to their complex architecture and the vast number of hyperparameters involved. This process is time consuming, requires computational resources and cautious fine-tuning to balance performance and efficiency.

Due to this extensive procedure, we chose to universally perform Informer model in the second case study, with the same hyperparameters. This involved using a large dataset containing 112 river basins and the model used the best set of hyperparameters, as revealed from the first implementation for a forecast horizon of 30 days. The upper had a totally different performance, resulting a coefficient of determination $R^2 = 0.4$ for all the stations (Figure 16 - left). Bad behaviour of Informer is represented also by the boxplot in Figure 16 (right), where for each station is calculated MAPE error. This case arises with a median value of 50%. An interesting insight of this large analysis is the fact that model performance is related to climacogram of the time series. Climacogram is a high effective stochastic tool [59] for detecting the long-term changes, thereby providing a better understanding of a process. We analysed stations that provide a wide range of different characteristics, as illustrated in Figure 19. We then tracked the MSE loss function values across all three subsets: training, validation, and test sets. An upward trend in MSE loss was observed as the Hurst coefficient H decreased. Figure 20 shows this outcome, where on x-axis each climacogram is sorted based on the slope, in ascending order. That means that climacogram with id near to zero indicates higher H value, which decreases as we move towards the right of the axis. From one point of view this observation is a bit anticipated, since a stochastic process with high Hurst value is less unpredictable in short-term prediction. The situation is being reversed in long-range prediction, where a time series with high Hurst coefficient has bigger uncertainty. In our case we had 30 days forecast horizon, which corresponds to short-term forecasting and results verify the above findings.

The scope of this study may be increased, leading to a major improvement in the methods employed in the current work to improve the performance and applicability of the models. To begin with, the development and evaluation of other transformer-based models and networks for the problem of streamflow forecasting should be considered. This great popularity they receive in other topics, such as natural language processing, suggests their likely utility in flood prediction as well. Additionally, in order to improve forecasts, the exploitation of other flood-related variables (e.g., temperature, rainfall, etc.) in the context of multivariate forecasting needs to be looked into. Another important aspect is quantifying the uncertainty of such models and addressing it through probabilistic forecasting. Moreover, the investigation of the performance of transformer ensemble modelling with statistical, stochastic, or other deep learning networks and the evaluation of models by implementing real-time predictions for immediate comparison can bring useful knowledge. Similarly to the approach taken in this research, an analysis of the impact that time series characteristics may have on the performance of the Informer model (and beyond) should be conducted [76], along with the identification of techniques or transformations to address any potential issues that arise. On top of that, post-processing techniques might boost the model's results. Lastly, but equally important, is the

attempt to interpret the operation of deep learning models (Interpretable AI) [77] and explain their results or the decisions made (Explainable AI) [78]. A way to achieve this is by incorporating models like PINNs, which integrate the underlying physics of the problem being addressed.

6. Summary and Conclusions

In this work, we study the contribution of deep learning networks to the prediction of time series related to flood events. Special emphasis was placed on modern deep learning tools such as transformer networks and in particular the Informer model. We executed two applications, where in the first one we compared four models on streamflow data from the River Test in England. Comparisons with more traditional deep learning networks (LSTM) and time series model (ARIMA) were included, along with the introduction of a naive Hurst-Kolmogorov stochastic approach for predictive modeling as a reference model. A distinct examination was conducted, as a secondary case study on the Informer model, evaluating its performance across a multitude of rivers in the UK. The analysis and processing of the applications led to some conclusions, which are summarised below:

- The stochastic approach is simple and theoretically substantiated on stochastic processes, suitable for use as a reference model.
- The technology of transformer based models shows great potential for improving the accuracy of time series forecasting models for both short-term and long-term sequences, when fine-tuning model's hyperparameters for each time series individually.
- The Informer model significantly decreases the time complexity and memory requirements compared to "vanilla" transformers. However the training and learning process remains a challenging and time-consuming task.
- Regarding the first application, the Informer model demonstrated superior performance in the River Test application, outperforming the other models. The ARIMA statistical model struggled to capture the complex patterns in the time series, when seasonality is not explicitly modeled. The LSTM network performed well for short-term forecasting, which is not the case for the long-term prediction, whereas Informer delivered better results.
- In the second case, exploring the Informer model across numerous rivers revealed that the performance of the model can be influenced by the structure of the time series data.
- Transformers come with drawbacks such as computational complexity, limited interpretability, and limited scalability. Therefore, each time series needs to be addressed and assessed independently.

Floods are a major problem nowadays and accurate forecasting helps in dealing with them. However, it remains a challenging task to solve. Overall transformer's architecture is notably promising in addressing complex temporal patterns and concepts, which is the growing consensus. However, as has been noted, computational complexity and scalability pose limitations that need to be addressed. More analysis on those kinds of networks should be explored in future hydrologic research. Future research should focus on optimizing these models to enhance efficiency and interpretability, ensuring their practical applicability in flood forecasting, as well as exploring the impact of different hyperparameters on their performance.

Author Contributions: Conceptualization, N.T.; methodology, N.T.; software, N.T.; validation, N.T.; formal analysis, N.T.; investigation, N.T., D.K., T.I. and P.D.; resources, N.T.; data curation, N.T.; writing—original draft preparation, N.T.; writing—review and editing, N.T. and D.K.; visualization, N.T.; supervision, N.T., D.K., T.I. and P.D.; project administration, N.T., D.K., T.I. and P.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data used, were retrieved from UK National River Flow Archive and are available online for free (<https://nrfa.ceh.ac.uk/data/search> (accessed on 10 February 2023)). The data are provided under the terms and conditions specified by the archive.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yu, Q.; Wang, Y.; Li, N. Extreme Flood Disasters: Comprehensive Impact and Assessment. *Water* **2022**, *14* (8), 1211. <https://doi.org/10.3390/w14081211>.
2. Yan, Z.; Li, M. A Stochastic Optimization Model for Agricultural Irrigation Water Allocation Based on the Field Water Cycle. *Water* **2018**, *10* (8), 1031. <https://doi.org/10.3390/w10081031>.
3. Nearing, G. S.; Kratzert, F.; Sampson, A. K.; Pelissier, C. S.; Klotz, D.; Frame, J. M.; Prieto, C.; Gupta, H. V. What Role Does Hydrological Science Play in the Age of Machine Learning? *Water Resources Research* **2021**, *57* (3). <https://doi.org/10.1029/2020wr028091>.
4. Kumar, V.; Kedam, N.; Sharma, K. V.; Mehta, D. J.; Caloiero, T. Advanced Machine Learning Techniques to Improve Hydrological Prediction: A Comparative Analysis of Streamflow Prediction Models. *Water* **2023**, *15* (14), 2572. <https://doi.org/10.3390/w15142572>.
5. Liang, J.; Li, W.; Bradford, S.; Šimůnek, J. Physics-Informed Data-Driven Models to Predict Surface Runoff Water Quantity and Quality in Agricultural Fields. *Water* **2019**, *11* (2), 200. <https://doi.org/10.3390/w11020200>.
6. Wang, Z.-Y.; Qiu, J.; Li, F.-F. Hybrid Models Combining EMD/EEMD and ARIMA for Long-Term Streamflow Forecasting. *Water* **2018**, *10* (7), 853. <https://doi.org/10.3390/w10070853>.
7. Yang, M.; Wang, H.; Jiang, Y.; Lu, X.; Xu, Z.; Sun, G. GECA Proposed Ensemble-KNN Method for Improved Monthly Runoff Forecasting. *Water Resources Management* **2020**, *34* (2), 849–863. <https://doi.org/10.1007/s11269-019-02479-2>.
8. Ni, L.; Wang, D.; Singh, V. P.; Wu, J.; Wang, Y.; Tao, Y.; Zhang, J. Streamflow and rainfall forecasting by two long short-term memory-based models. *Journal of Hydrology* **2020**, *583*, 124296. <https://doi.org/10.1016/j.jhydrol.2019.124296>.
9. Ghimire, S.; Yaseen, Z. M.; Farooque, A. A.; Deo, R. C.; Zhang, J.; Tao, X. Streamflow prediction using an integrated methodology based on convolutional neural network and long short-term memory networks. *Scientific Reports* **2021**, *11* (1). <https://doi.org/10.1038/s41598-021-96751-4>.
10. Qi, X.; de Almeida; G. A.; Maldonado, S. Physics-informed neural networks for solving flow problems modeled by the 2D Shallow Water Equations without labeled data. *Journal of Hydrology* **2024**, *636*, 131263. <https://doi.org/10.1016/j.jhydrol.2024.131263>
11. Digital Science. *Dimensions AI: The Most Advanced Scientific Research Database*. Dimensions. (n.d.). Available online: <https://app.dimensions.ai/discover/publication> (accessed on 02 June 2024).
12. Box, G.E.P.; Jenkins, G.M. *Time Series Analysis: Forecasting and Control*; Holden-Day, San Francisco, 1970.
13. Ab Razak, N. H.; Aris, A. Z.; Ramli, M. F.; Looi, L. J.; Juahir, H. Temporal flood incidence forecasting for Segamat River (Malaysia) using autoregressive integrated moving average modelling. *Journal of Flood Risk Management* **2016**, *11*, S794–S804. <https://doi.org/10.1111/jfr3.12258>.
14. Nourani, V.; Najafi, H.; Amini, A. B.; Tanaka, H. Using Hybrid Wavelet-Exponential Smoothing Approach for Streamflow Modeling. *Complexity* **2021**, *2021*, 1–17. <https://doi.org/10.1155/2021/6611848>.
15. Valipour, M.; Banihabib, M. E.; Behbahani, S. M. R. Comparison of the ARMA, ARIMA, and the autoregressive artificial neural network models in forecasting the monthly inflow of Dez dam reservoir. *Journal of Hydrology* **2013**, *476*, 433–441. <https://doi.org/10.1016/j.jhydrol.2012.11.017>.
16. Valipour, M.; Banihabib, M. E.; & Behbahani; S. M. R. Parameters Estimate of Autoregressive Moving Average and Autoregressive Integrated Moving Average Models and Compare Their Ability for Inflow Forecasting. *Journal of Mathematics and Statistics* **2012**, *8* (3), 330–338. <http://dx.doi.org/10.3844/jmssp.2012.330.338>.
17. Jozaghi, A.; Shen, H.; Ghazvinian, M.; Seo, D.-J.; Zhang, Y.; Welles, E.; Reed, S. Multi-model streamflow prediction using conditional bias-penalized multiple linear regression. *Stochastic Environmental Research and Risk Assessment* **2021**, *35* (11), 2355–2373. <https://doi.org/10.1007/s00477-021-02048-3>.
18. Yaseen, Z. M.; El-shafie, A.; Jaafar, O.; Afan, H. A.; Sayl, K. N. Artificial intelligence based models for stream-flow forecasting: 2000–2015. *Journal of Hydrology* **2015**, *530*, 829–844. <https://doi.org/10.1016/j.jhydrol.2015.10.038>.
19. Ikram, R. M. A.; Ewees, A. A.; Parmar, K. S.; Yaseen, Z. M.; Shahid, S.; Kisi, O. The viability of extended marine predators algorithm-based artificial neural networks for streamflow prediction. *Applied Soft Computing* **2022**, *131*, 109739. <https://doi.org/10.1016/j.asoc.2022.109739>.

20. Noori, R.; Karbassi, A. R.; Moghaddamnia, A.; Han, D.; Zokaei-Ashtiani, M. H.; Farokhnia, A.; Gousheh, M. G. Assessment of input variables determination on the SVM model performance using PCA, Gamma test, and forward selection techniques for monthly stream flow prediction. *Journal of Hydrology* **2011**, *401* (3-4), 177–189. <https://doi.org/10.1016/j.jhydrol.2011.02.021>.
21. Papacharalampous, G. A.; Tyrallis, H. Evaluation of random forests and Prophet for daily streamflow forecasting. *Advances in Geosciences* **2018**, *45*, 201–208. <https://doi.org/10.5194/adgeo-45-201-2018>.
22. Ni, L.; Wang, D.; Wu, J.; Wang, Y.; Tao, Y.; Zhang, J.; Liu, J. Streamflow forecasting using extreme gradient boosting model coupled with Gaussian mixture model. *Journal of Hydrology* **2020**, *586*, 124901. <https://doi.org/10.1016/j.jhydrol.2020.124901>.
23. Szczepanek, R. Daily Streamflow Forecasting in Mountainous Catchment Using XGBoost, LightGBM and CatBoost. *Hydrology* **2022**, *9* (12), 226. <https://doi.org/10.3390/hydrology9120226>.
24. Chang, F.-J.; Chang, L.-C.; Chen, J.-F. Artificial Intelligence Techniques in Hydrology and Water Resources Management. *Water* **2023**, *15* (10), 1846. <https://doi.org/10.3390/w15101846>.
25. Sahoo, B. B.; Jha, R.; Singh, A.; Kumar, D. Long short-term memory (LSTM) recurrent neural network for low-flow hydrological time series forecasting. *Acta Geophysica* **2019**, *67* (5), 1471–1481. <https://doi.org/10.1007/s11600-019-00330-1>.
26. Zhao, X.; Lv, H.; Wei, Y.; Lv, S.; Zhu, X. Streamflow Forecasting via Two Types of Predictive Structure-Based Gated Recurrent Unit Models. *Water* **2021**, *13* (1), 91. <https://doi.org/10.3390/w13010091>.
27. Shu, X.; Ding, W.; Peng, Y.; Wang, Z.; Wu, J.; Li, M. Monthly Streamflow Forecasting Using Convolutional Neural Network. *Water Resources Management* **2021**, *35*, 5089–5104. <https://doi.org/10.1007/s11269-021-02961-w>.
28. Salinas, D.; Flunkert, V.; Gasthaus, J.; Januschowski, T. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* **2019**, *36* (3). <https://doi.org/10.1016/j.ijforecast.2019.07.001>.
29. Taylor, S. J.; Letham, B. Forecasting at Scale. *The American Statistician* **2018**, *72* (1), 37–45. <https://doi.org/10.1080/00031305.2017.1380080>.
30. Zou, Y.; Wang, J.; Peng, L.; Li, Y. A novel multi-step ahead forecasting model for flood based on time residual LSTM. *Journal of Hydrology* **2023**, *620*, 129521. <https://doi.org/10.1016/j.jhydrol.2023.129521>.
31. Tyrallis, H.; Papacharalampous, G. A. Large-scale assessment of Prophet for multi-step ahead forecasting of monthly streamflow. *Advances in Geosciences* **2018**, *45*, 147–153. <https://doi.org/10.5194/adgeo-45-147-2018>.
32. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Advances in neural information processing systems* **2017**, *30*, 5998–6008.
33. Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; Sun, L. Transformers in Time Series: A Survey. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI'23)* **2022**.
34. Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y. X.; Yan, X. Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. *Advances in neural information processing systems* **2019**, *32*.
35. Wu, N.; Green, B.; Ben, X.; O'Banion, S. Deep Transformer Models for Time Series Forecasting: The Influenza Prevalence Case. *arXiv preprint* **2020**, arXiv:2001.08317.
36. Wang, S.; Li, B. Z.; Khabsa, M.; Fnag, H.; Ma, H. Linformer: Self-Attention with Linear Complexity. *arXiv preprint* **2020**, arXiv:2006.04768.
37. Kitaev, N.; Kaiser, L.; Levskaya, A. Reformer: The Efficient Transformer. In *International Conference on Learning Representations* **2020**. <https://openreview.net/forum?id=rkgNkkHtvB>.
38. Beltagy, I.; Peters, M. E.; and Cohan, A. Longformer: The Long-Document Transformer. *arXiv preprint* **2020**, arXiv:2004.05150.
39. Yang, L.; Koprinska, I.; Rana, M. SSDNet: State Space Decomposition Neural Network for Time Series Forecasting. *2021 IEEE International Conference on Data Mining (ICDM)*, Auckland, New Zealand, **2021**, 370–378. <https://doi.org/10.1109/icdm51629.2021.00048>.
40. Lim, B.; Arik, S. Ö.; Loeff, N.; Pfister, T. Temporal Fusion Transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting* **2021**, *37* (4), 1748–1764. <https://doi.org/10.1016/j.ijforecast.2021.03.012>.

41. Wu, H.; Xu, J.; Wang, J.; Long, M. Autoformer: Decomposition Transformers with Auto-Correlation for Long-Term Series Forecasting. In *Proceedings of the Advances in Neural Information Processing Systems (NeurIPS) 2021*, 34, 22419-22430.
42. Cirstea, R. G.; Guo, C.; Yang, B.; Kieu, T.; Dong, X.; Pan, S. Triformer: Triangular, Variable-Specific Attentions for Long Sequence Multivariate Time Series Forecasting. *arXiv preprint 2022*, arXiv:2204.13767.
43. Shen, L.; Wang, Y. TCCT: Tightly-coupled convolutional transformer on time series forecasting. *Neurocomputing 2022*, 480, 131-145. <https://doi.org/10.1016/j.neucom.2022.01.039>.
44. Liu, S.; Yu, H.; Liao, C.; Li, J.; Lin, W.; Liu, A. X.; Dustdar, S. Pyraformer: Low-complexity pyramidal attention for long-range time series modeling and forecasting. In *International Conference on Learning Representations 2022*.
45. Drouin, A.; Marcotte, É.; Chapados, N. Tactis: Transformer-attentional copulas for time series. In *Proceedings of the 39th International Conference on Machine Learning 2022*, 5447-5493, PMLR 162:5447-5493.
46. Zhou, T.; Ma, Z.; Wen, Q.; Wang, X.; Sun, L.; Jin, R. FEDformer: Frequency Enhanced Decomposed Transformer for Long-term Series Forecasting. In *39th International Conference on Machine Learning (ICML) 2022*, 27268-27286, PMLR 162:27268-27286.
47. Shabani, A.; Abdi, A.; Meng, L.; Sylvain, T. Scaleformer: iterative multi-scale refining transformers for time series forecasting. *arXiv preprint 2022*, arXiv:2206.04038.
48. Zhang, Y.; Yan, J. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *The Eleventh International Conference on Learning Representations, 2023*.
49. Fayer, G.; Lima, L.; Miranda, F.; Santos, J.; Campos, R.; Bignoto, V.; ... Goliatt, L. A Temporal Fusion Transformer Deep Learning Model for Long-Term Streamflow Forecasting: A Case Study in the Funil Reservoir, Southeast Brazil. *Knowledge-Based Engineering and Sciences 2023*, 4 (2), 73-88.
50. Ghobadi, F.; Kang, D. Improving long-term streamflow prediction in a poorly gauged basin using geospatiotemporal mesoscale data and attention-based deep learning: A comparative study. *Journal of Hydrology 2022*, 615, 128608. <https://doi.org/10.1016/j.jhydrol.2022.128608>.
51. Liu, C.; Liu, D.; Mu, L. Improved Transformer Model for Enhanced Monthly Streamflow Predictions of the Yangtze River. *IEEE Access 2022*, 10, 58240-58253. <https://doi.org/10.1109/access.2022.3178521>.
52. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence 2021*, 35 (12), 11106-11115, <https://doi.org/10.1609/aaai.v35i12.17325>.
53. Shang, J.; Zhao, B.; Hua, H.; Wei, J.; Qin, G.; Chen, G. Application of Informer Model Based on SPEI for Drought Forecasting. *Atmosphere 2023*, 14 (6), 951. <https://doi.org/10.3390/atmos14060951>.
54. Yang, Z.; Liu, L.; Li, N.; Tian, J. Time Series Forecasting of Motor Bearing Vibration Based on Informer. *Sensors 2022*, 22 (15), 5858. <https://doi.org/10.3390/s22155858>.
55. Huang, X.; Jiang, A. Wind Power Generation Forecast Based on Multi-Step Informer Network. *Energies 2022*, 15 (18), 6642. <https://doi.org/10.3390/en15186642>.
56. Wei, H.; Wang, Y.; Liu, J.; Cao, Y. Monthly Runoff Prediction by Combined Models Based on Secondary Decomposition at the Wulong Hydrological Station in the Yangtze River Basin. *Water 2023*, 15 (21), 3717. <https://doi.org/10.3390/w15213717>.
57. Van Erven, T.; Harremos, P. Rényi Divergence and Kullback-Leibler Divergence. *IEEE Transactions on Information Theory 2014*, 60 (7), 3797-3820. <https://doi.org/10.1109/tit.2014.2320500>.
58. Koutsoyiannis, D. Hydrology and change. *Hydrological Sciences Journal 2013*, 58 (6), 1177-1197. <https://doi.org/10.1080/02626667.2013.804626>.
59. Dimitriadis, P.; Koutsoyiannis, D. Climacogram versus autocovariance and power spectrum in stochastic modelling for Markovian and Hurst-Kolmogorov processes. *Stochastic Environmental Research & Risk Assessment 2015*, 29 (6), 1649-1669. <https://doi.org/10.1007/s00477-015-1023-7>.
60. Koutsoyiannis, D. HESS Opinions" A random walk on water". *Hydrology and Earth System Sciences 2010*, 14 (3), 585-601. <https://doi.org/10.5194/hess-14-585-2010>.
61. Koutsoyiannis, D. Generic and parsimonious stochastic modelling for hydrology and beyond. *Hydrological Sciences Journal 2016*, 61 (2), 225-244. <https://doi.org/10.1080/02626667.2015.1016950>.
62. Hurst, H. E. Long-Term Storage Capacity of Reservoirs. *Transactions of the American Society of Civil Engineers 1951*, 116 (1), 770-799. <https://doi.org/10.1061/taceat.0006518>.

63. Koutsoyiannis, D. *Stochastics of Hydroclimatic Extremes - A Cool Look at Risk*, 3rd ed; Kallipos Open Academic Editions, Athens, 2022; 346 pages, ISBN: 978-618-85370-0-2. <http://dx.doi.org/10.57713/kallipos-1>.
64. Akaike, H. Akaike's information criterion. *International encyclopedia of statistical science* **2011**, 25.
65. Rumelhart, D. E.; Hinton, G. E., Williams, R. J. Learning Internal Representations by Error Propagation. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, MIT Press, **1987**, 318-362.
66. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Computation* **1997**, 9 (8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>.
67. Hochreiter, S. The Vanishing Gradient Problem during Learning Recurrent Neural Nets and Problem Solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* **1998**, 6 (02), 107–116. <https://doi.org/10.1142/s0218488598000094>.
68. Olah, C. *Understanding LSTM Networks*. Colah's Blog. Available online: <https://colah.github.io/posts/2015-08-Understanding-LSTMs/> (accessed on 28 February 2023).
69. Zhou, H., (2021). *Informer2020*. *GitHub*. Available online: <https://github.com/zhouhaoyi/Informer2020> (accessed on 05 August 2023).
70. Clevert, D. A.; Unterthiner, T.; Hochreiter, S. Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs). *arXiv preprint* **2015**, arXiv:1511.07289.
71. River Test. Wikipedia. Available online: https://en.wikipedia.org/wiki/River_Test (accessed on 15 June 2023).
72. Met Office (2006): UK Daily Temperature Data, Part of the Met Office Integrated Data Archive System (MIDAS). NCAS British Atmospheric Data Centre. <https://data.ceda.ac.uk/badc/ukmo-hadobs/data/insitu/MOHC/HadOBS/HadUK-Grid/v1.1.0.0> (accessed on 10 February 2023).
73. Department for Environment Food & Rural Affairs. API documentation. <https://environment.data.gov.uk/hydrology/doc/reference> (accessed on 10 February 2023).
74. UK National River Flow Archive. <https://nrfa.ceh.ac.uk/> (accessed on 10 February 2023).
75. Amanambu, A. C.; Mossa, J.; Chen, Y.-H. Hydrological Drought Forecasting Using a Deep Transformer Model. *Water* **2022**, 14 (22), 3611. <https://doi.org/10.3390/w14223611>.
76. Solís, M.; Calvo-Valverde, L.-A. Explaining When Deep Learning Models Are Better for Time Series Forecasting. *Engineering Proceedings* **2024**, 68 (1), 1. <https://doi.org/10.3390/engproc2024068001>.
77. Dikshit, A.; Pradhan, B. Interpretable and explainable AI (XAI) model for spatial drought prediction. *Science of the Total Environment* **2021**, 801, 149797. <https://doi.org/10.1016/j.scitotenv.2021.149797>.
78. Linardatos, P.; Papastefanopoulos, V.; Kotsiantis, S. Explainable AI: A Review of Machine Learning Interpretability Methods. *Entropy* **2020**, 23 (1), 18. <https://doi.org/10.3390/e23010018>.
79. Author 1, T. The title of the cited article. *Journal Abbreviation* **2008**, 10, 142–149.
80. Author 2, L. The title of the cited contribution. In *The Book Title*; Editor 1, F., Editor 2, A., Eds.; Publishing House: City, Country, 2007; pp. 32–58.
81. Author 1, A.; Author 2, B. *Book Title*, 3rd ed.; Publisher: Publisher Location, Country, 2008; pp. 154–196.
82. Author 1, A.B.; Author 2, C. Title of Unpublished Work. *Abbreviated Journal Name* year, phrase indicating stage of publication (submitted; accepted; in press).
83. Author 1, A.B. (University, City, State, Country); Author 2, C. (Institute, City, State, Country). Personal communication, 2012.
84. Author 1, A.B.; Author 2, C.D.; Author 3, E.F. Title of presentation. In Proceedings of the Name of the Conference, Location of Conference, Country, Date of Conference (Day Month Year); Abstract Number (optional), Pagination (optional).
85. Author 1, A.B. Title of Thesis. Level of Thesis, Degree-Granting University, Location of University, Date of Completion.
86. Title of Site. Available online: URL (accessed on 11 July 2023).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.