scientific reports

OPEN

Temporal neural operator for modeling time-dependent physical phenomena

Waleed Diab[1] & Mohammed Al Kobaisi[1,2]✉

Neural Operators (NOs) are machine learning models designed to solve partial differential equations (PDEs) by learning to map between function spaces. Neural Operators such as the Deep Operator Network (DeepONet) and the Fourier Neural Operator (FNO) have demonstrated excellent generalization properties when mapping between spatial function spaces. However, they struggle in mapping the temporal dynamics of time-dependent PDEs, especially for time steps not explicitly seen during training. This limits their temporal accuracy as they do not leverage these dynamics in the training process. In addition, most NOs tend to be prohibitively costly to train, especially for higher-dimensional PDEs. In this paper, we propose the Temporal Neural Operator (TNO), an efficient neural operator specifically designed for spatio-temporal operator learning for time-dependent PDEs. TNO achieves this by introducing a temporal-branch to the DeepONet framework, leveraging the best architectural design choices from several other NOs, and a combination of training strategies including Markov assumption, teacher forcing, temporal bundling, and the flexibility to condition the output on the current state or past states. Through extensive benchmarking and an ablation study on a diverse set of example problems we demonstrate the TNO long range temporal extrapolation capabilities, robustness to error accumulation, resolution invariance, and flexibility to handle multiple input functions.

Keywords Neural operators, Scientific machine learning, Climate modeling, Weather forecast, Carbon sequestration

Time-dependent Partial Differential Equations (PDEs) are fundamental tools for modeling the dynamic behavior of complex physical, biological, and environmental systems, including fluid dynamics, heat transfer, wave propagation, and biological processes. Despite their widespread applicability, significant challenges hinder their use in real-world scenarios. Many problems lack analytical solutions, necessitating computationally intensive numerical simulations. Moreover, incomplete or uncertain knowledge of the underlying physics can lead to inaccuracies in model formulation. In domains such as climate and weather forecasting, the availability of vast observational data presents both an opportunity and a challenge: integrating large-scale heterogeneous datasets–often marred by noise, gaps, and inconsistencies–into high-fidelity simulations requires advanced data assimilation techniques and substantial computational resources[1,2]. These limitations underscore the need for alternative approaches that can effectively leverage available data while mitigating computational and modeling burdens.

In recent years, machine learning for scientific computing applications has gained wide spread popularity due to its high potency in handling large volumes of data. Physics-Informed Neural Networks (PINNs)[3] have been introduced as a deep learning framework that can learn nonlinear dynamics by incorporating physical laws, expressed as partial differential equations (PDEs), directly into the loss function; thus ensuring that the model adheres to the governing physical principles while fitting observed data. However, PINNs have not shown significant computational gains compared to traditional numerical methods, particularly when dealing with large-scale simulations or stiff systems, where the iterative nature of training deep neural networks can result in slower run-times and higher resource consumption[4–6]. Neural operator learning, which aims to learn mappings between function spaces rather than individual point-wise solutions, was later introduced through the Deep Operator Network (DeepONet)[7]. This approach enables models to generalize more effectively across varying inputs and provides a significant advantage in handling diverse physical systems. Architectures like the DeepONet[7] and the Fourier Neural Operator (FNO)[8] exemplify this approach. These architectures have

[1]Chemical and Petroleum Engineering, Khalifa University, Abu Dhabi, United Arab Emirates. [2]Delft Institute of Applied Mathematics, Delft University of Technology, Delft, The Netherlands. ✉email: mohammed.alkobaisi@ku.ac.ae; AlKobaisi@tudelft.nl

demonstrated superior scalability and computational efficiency when solving complex partial differential equations, especially in large-scale and high-dimensional problems.

While neural operators[7–13] provide a promising approach to address the challenges of solving time-dependent PDEs, they often struggle with extrapolation beyond the temporal training horizon[10,14–17]. This limitation hampers their effectiveness for long-term predictions, as performance typically degrades when forecasting solutions beyond the temporal range of the training set. Moreover, these architectures often rely on fixed spatio-temporal grids which further reduces their flexibility in handling variable resolutions.

In this work, we introduce the Temporal Neural Operator (TNO), a novel neural operator architecture that enables simultaneous generalization to new PDE instances and temporal extrapolation beyond the training dataset with negligible error accumulation over time. The TNO extends the DeepONet framework by introducing a novel temporal branch and by incorporating architectural elements and training strategies inspired by prior Neural Operator (NO) research[7,8,10,18,19]. The TNO architecture includes several design components that contribute to its performance. First, the branch network of the TNO includes an encoder layer, a processing layer, and a nonlinear activation. The encoder is inspired by the Fourier Neural Operator (FNO)[8], where a linear transformation is used to lift the input functions into a higher-dimensional latent space. This lifting step is critical, as the raw input function space may not sufficiently capture the complexity of the operator learning domain. The subsequent processor layer serves as a feature extractor that enables the network to learn a compact representation of the target solution space. This processor may take various forms, including feedforward neural networks (FNNs), recurrent neural networks (RNNs), long short-term memory (LSTM) networks, convolutional neural networks (CNNs), U-Nets, or spectral/Fourier layers. The choice of architecture is problem-dependent; in this study, we consistently employ a U-Net architecture due to its proven efficacy in feature extraction[9,20–24]. Second, the trunk network is implemented as an FNN to provide a continuous representation over the spatio-temporal domain. This continuity enables interpolation in time and supports temporal super-resolution. This design choice aligns with earlier work on operator learning over continuous input spaces[7,10]. Third, inspired by the Multiple-input Operator Network (MIONet)[18], we introduce a temporal branch (t-branch) that explicitly models the temporal dynamics of the system. Like the branch, the t-branch consists of an encoder, a processor, and an activation layer. The t-branch architecture may differ from that of the branch, provided the latent representations produced are of compatible dimensions to permit a valid Hadamard (element-wise) product during fusion. Finally, a decoder maps the fused latent representation back to the output solution space. This component is also influenced by the FNO[8] design. Although we employ an FNN decoder to maintain continuity in the output field, alternative choices such as CNNs may achieve comparable performance depending on the application.

To enable resolution-agnostic training and inference, we apply downsampling to the input fields before processing them with the U-Net/CNN and then apply corresponding upsampling to the output. These operations allow the TNO to handle data across a range of discretizations and to perform super-resolution prediction. We use interpolation-based schemes (e.g., bilinear) to preserve spatial smoothness and reduce aliasing artifacts. To further mitigate this issue, we note that the initial condition provided to the TNO is given at the target output resolution. As a result, the network directly predicts the temporal evolution at the desired spatial scale, significantly reducing the risk of losing fine-scale structures during downsampling or upsampling.

Building on Brandstetter et al.[19], we apply temporal bundling, whereby the model predicts a block of future timesteps in a single forward pass. Bundling cuts the number of inference steps, which limits cumulative distribution shifts and curbs error growth over long roll-outs. During training, we optionally employ *teacher forcing*, a sequence learning strategy in which the model is provided with the ground-truth values of previous timesteps rather than its own predictions. This technique helps stabilize learning and reduces error accumulation in autoregressive rollout. In combination, these architectural and training strategies enable the TNO to effectively capture both spatial and temporal dependencies, while improving computational efficiency, accelerating training, and reducing long-term prediction error. This results in a flexible and scalable framework for learning parametric solution operators of time-dependent PDEs.

In this work, we propose a simple trick that enables the TNO to learn 3D problems by conditioning on one spatial axis (e.g., depth or level), which allows it to learn 2D slices independently using 2D operations. This approach leads to significant reductions in training cost. This trick is demonstrated in our first example of a 3D global daily air temperature prediction, along with long-term temporal extrapolation. In the second example, we showcase the TNO's zero-shot super-resolution capability using historical daily observational air temperatures over Europe. Here, the TNO is trained on historical air temperature data at a 0.25° grid resolution, and tested on future air temperature data at a 0.1° grid resolution. The TNO demonstrates superb performance in the simultaneous temporal extrapolation and super-resolution with high accuracy and error invariance to resolution. In the third example, we validate the TNO's ability to generalize across diverse input functions and variables on a carbon sequestration dataset which tracks both saturation and pressure buildup. Despite the limited number of available time steps, the diversity of PDE variables, and the complexity of the problem, the TNO achieves robust temporal extrapolation and generalization performance. The three examples presented herein encapsulate the versatile potential of TNO to tackle complex, time-dependent real-world problems. Overall, we make the following key contributions:

- *Architecture* We introduce the temporal-branch (t-branch), encoder layers and U-Net blocks in the branch and t-branch, we replace the dot product with the Hadamard product, and introduce a decoder Feed-Forward Neural Network (FFN).
- *Training* We devise a flexible training strategy with temporal-bundling to encourage stable long rollouts into the future and reduce error accumulation, as well as teacher forcing for accelerated training.

- *Flexibility* The TNO is a flexible framework which can be conditioned autoregressively or on a memory of past states.
- *Efficiency* We demonstrate how to utilize the strong generalization capabilities of the TNO to model 3D problems using 2D operations.

The combination of strong generalization to multiple new PDE parameters, long accurate temporal extrapolation with minimal error accumulation, flexible input and output step size, in addition to the TNO low memory footprint, and super resolution capabilities, positions the TNO as the State-Of-The-Art in neural operator learning for time dependent PDEs. To strengthen this claim, the TNO capabilities are demonstrated on three real-world challenging problems.

Time-dependent partial differential equations

Time-dependent partial differential equations (PDEs) describe the spatio-temporal evolution of a solution, denoted as $u(t, \mathrm{x})$, across a temporal domain $t \in [0, T]$ and a spatial domain $\mathrm{x} = [x_1, x_2, ..., x_m] \in \mathcal{X} \subseteq \mathbb{R}^m$, where *m* is the spatial dimension. We allow $u(t, \mathrm{x}) \in \mathbb{R}^{n'}$ to be either scalar- or vector-valued (so $n' = 1$ in the scalar case). These PDEs relate the temporal derivative u_t to the spatial derivatives u_x, $u_{\mathrm{xx},...}$, through a general functional form *F*, such that:

$$u_t = F(t, \mathrm{x}, u, u_\mathrm{x}, u_\mathrm{xx}, ...). \tag{1}$$

One way for analyzing the temporal dynamics of these systems is offered by operator learning, which focuses on identifying mappings between function spaces. In this context, we define a time evolution operator $\mathcal{G}_{\Delta t} : \mathbb{R}_{>0} \times \mathbb{R}^n \to \mathbb{R}^{n'}$, where *n* is the number of spatial degrees of freedom at a given time after discretization, and n' is the dimensionality of the output (often $n' = n$, but not necessarily). Specifically, the solution at a future time $t + \Delta t$ is obtained by applying this operator to the spatial profile of the solution at the current time *t*, denoted as $u(t, \cdot)$:

$$u(t + \Delta t) = \mathcal{G}_{\Delta t}(t, u(t, \cdot)). \tag{2}$$

The operator $\mathcal{G}$ acts on function spaces, mapping $u \in \mathcal{U}$ to $u' \in \mathcal{U}'$, where the input function $u \in \mathcal{U}$ is mapped to the output function $u(t + \Delta t) \in \mathcal{U}'$. Here, $u : \mathcal{X} \to \mathbb{R}^n$, $u' : \mathcal{X}' \to \mathbb{R}^{n'}$, and the spatial domain is defined such that $\mathcal{X} \in \mathbb{R}^m$, and $\mathcal{X}' \in \mathbb{R}^{m'}$. Equation 2 implicitly assumes a Markov property for neural operators, where future state of the system depends only on the current state and not on the history of past states. As such, we say that the neural operator is autoregressively parametrized. Alternatively, the neural operator can be conditioned on the memory of past system states. Consequently, equation 2 can be extended to an arbitrary number of input time steps *L* and output time steps *K*, where *K* is the temporally bundled predictions[19].

Extending the single-step formulation, we define a history of *L* past states of the solution as

$$\mathrm{U}_{hist}(t) = \{u(t - (l - 1)\Delta t, \cdot)\}_{l=1}^{L}, \tag{3}$$

and a bundle of *K* future states as

$$\mathrm{U}_{fut}(t) = \{u(t + k\Delta t, \cdot)\}_{k=1}^{K}. \tag{4}$$

The extended time evolution operator, denoted by $\mathcal{G}_{\Delta t}^{L \to K}$, maps the input history to the sequence of future states:

$$\mathrm{U}_{fut}(t) = \mathcal{G}_{\Delta t}^{L \to K}(t, \mathrm{U}_{hist}(t)). \tag{5}$$

More explicitly, this can be written as:

$$\begin{pmatrix} u(t + \Delta t, \cdot) \\ u(t + 2\Delta t, \cdot) \\ \vdots \\ u(t + K\Delta t, \cdot) \end{pmatrix} = \mathcal{G}_{\Delta t}^{L \to K} \left(t, \begin{pmatrix} u(t - (L - 1)\Delta t, \cdot) \\ \vdots \\ u(t - \Delta t, \cdot) \\ u(t, \cdot) \end{pmatrix} \right).$$

Many architectures[25–29] approach neural operator learning of time dependent PDEs by utilizing a memory module to keep a compressed memory of the system past states, which is akin to choosing $L > 1$. On the other hand, choosing $L = 1$ regains the autoregressive approach found in many other architectures[11,12,30]. Choosing $L = 1$ or $L > 1$ is problem specific, however, we found that it is always a good idea to choose $K > 1$ as it reduces the number of solution calls. Moreover, $K = K_{max}$, that is, predicting all available time steps at once may cause a neural operator to lose its ability to learn temporal dynamics, and subsequently the temporal extrapolation ability.

The TNO learns the time evolution operator from the data and effectively predicts the future states of the system governed by the PDE without explicit knowledge of the underlying differential equation *F*. This data-driven approach provides a direct pathway for forecasting the temporal behavior of PDE solutions based on observed or computed past states. In theory, a well-trained model can be rolled-out indefinitely; in practice however, error accumulation with excessive successive roll-outs render most temporal neural operators useless after a relatively short time horizon. Alternatively, one can obtain predictions for longer time horizons by training a temporal operator for large Δt, which can limit the accumulation of errors, but does not eliminate it entirely.

Temporal neural operator (TNO)

The Deep Operator Network (DeepONet)[7] consists of two primary components: a branch network (b_i), which processes input functions, and a trunk network (tT_i), which processes query locations. The learned operator $\mathcal{G}_\theta$ approximates a nonlinear operator that maps an input function v to its output at a query point y, and is expressed as:

$$\mathcal{G}_\theta(v)(y) = \sum_{i=1}^{p} b_i \cdot T_i = \sum_{i=1}^{p} b_i(v(x_1), v(x_2), \ldots, v(x_m)) \cdot T_i(y). \tag{6}$$

The branch network takes as input a discretized representation of the function v, evaluated at a set of predefined sensor points $\{x_i\}_{i=1}^m$. This results in the input vector $[v(x_1), v(x_2), \ldots, v(x_m)]^T$, which the branch network maps to a feature vector $[b_1, b_2, \ldots, b_p]^T \in \mathbb{R}^p$. The trunk network takes the query location y as input and outputs another feature vector $[t_1, t_2, \ldots, t_p]^T \in \mathbb{R}^p$. Here, p is the latent dimension of the learned representation space. The output of the DeepONet is the inner product of these two feature vectors which yields $\mathcal{G}_\theta(v)(y)$.

We extend the DeepONet framework by introducing a temporal branch (t-branch) that processes solution snapshots $u(t, \cdot)$. This t-branch is designed to capture the temporal dynamics of the system and, when trained using temporal bundling, enables the model to effectively interpolate and extrapolate in time. The output of the proposed Temporal Neural Operator (TNO) is the predicted sequence of future states $\hat{U}_{fut}$, computed as a nonlinear function of the Hadamard product (element-wise multiplication) of the branch, trunk, and t-branch outputs. This combined representation is then projected into the output solution space of temporal length K. This architecture generalizes the DeepONet by incorporating temporal dynamics directly into the operator learning framework:

$$\mathcal{G}_\theta^{L\to K}(\mathrm{U}_{\mathrm{hist}}(t))(x) = G\left(b_i(v) \odot T_i(t,x) \odot tb_i(\mathrm{U}_{\mathrm{hist}}(t))\right), \quad \text{where } b_i, T_i, tb_i \in \mathbb{R}^p \tag{7}$$

where G is the projection to the solution space of size K in time. Here, p denotes the latent dimension of the learned representation space, shared across the outputs of the branch, trunk, and temporal-branch networks. The Hadamard product combines these three p-dimensional feature vectors into a unified representation, which is then mapped by the projection operator $G : \mathbb{R}^p \to \mathbb{R}^{K\times d}$, parametrized as a multilayer perceptron (MLP), to predict the future solution sequence $\widehat{\mathrm{U}}_{\mathrm{fut}}(t)$.

As an example, with $L = 1$ and $K = 3$, we illustrate how the TNO operates at a given time step t_0. The input to the TNO at time t_0 consists of the triplet $\{u(t_0, \cdot),\ (t_0, \cdot),\ v(t_0, \cdot)\}$, where $(t_0, \cdot)$ represents the full spatio-temporal coordinate field, and $v(t_0, \cdot)$ denotes an auxiliary time-dependent input function (e.g., a forcing term or coefficient field). The network predicts a bundle of future solution states $\{\hat{u}(t_1, \cdot),\ \hat{u}(t_2, \cdot),\ \hat{u}(t_3, \cdot)\}$; see Fig. 1 for a visual summary of the training procedure using temporal bundling. Note that this example is used solely for illustrative purposes. In practice, the values of L and K are chosen empirically and vary across benchmark problems based on prior knowledge of the system's temporal dynamics, desired prediction horizon, and computational considerations. This flexibility highlights a key strength of the TNO architecture, which can adapt to different problem settings and rollout strategies.

Temporal bundling enables the TNO to produce multiple future steps in a single forward pass, thereby improving efficiency and training stability. Once the initial bundle is predicted, the final output $\hat{u}(t_3, \cdot)$ is reused

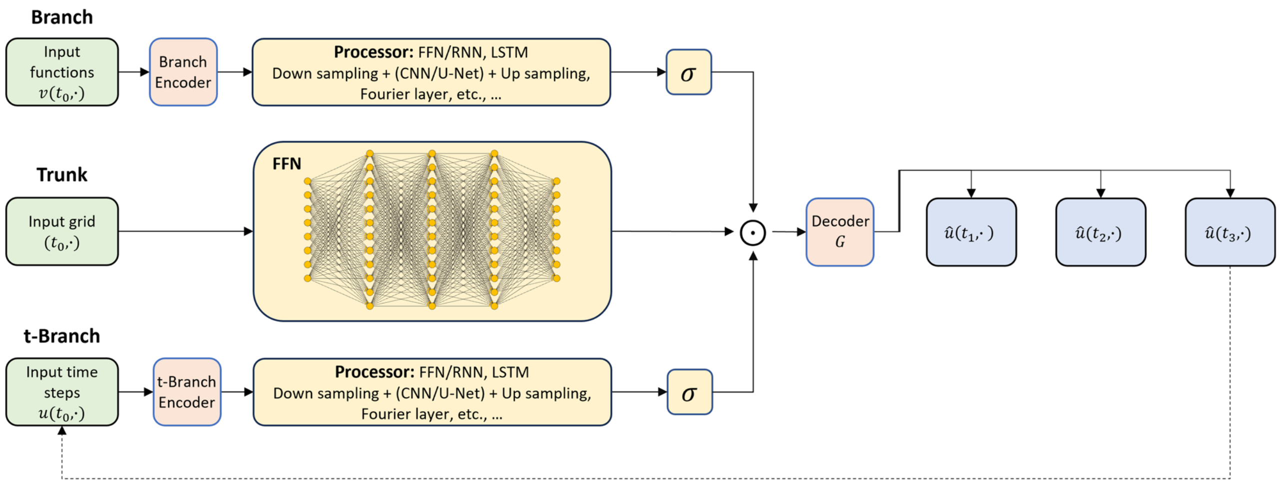

Branch
Input functions $v(t_0,\cdot)$
Branch Encoder
Processor: FFN/RNN, LSTM Down sampling + (CNN/U-Net) + Up sampling, Fourier layer, etc., ...
σ
Trunk
Input grid $(t_0,\cdot)$
FFN
⊙
Decoder G
$\hat{u}(t_1,\cdot)$
$\hat{u}(t_2,\cdot)$
$\hat{u}(t_3,\cdot)$
t-Branch
Input time steps $u(t_0,\cdot)$
t-Branch Encoder
Processor: FFN/RNN, LSTM Down sampling + (CNN/U-Net) + Up sampling, Fourier layer, etc., ...
σ

Fig. 1. TNO architecture with temporal bundling training procedure. In this figure, $L = 1$ and $K = 3$ as an example.

as part of the next input for autoregressive rollout. During training, if teacher forcing is employed, the ground truth $u(t_3, \cdot)$ is used instead to stabilize learning and mitigate error accumulation.

The architecture of the TNO is illustrated in Fig. 1 and described in detail next. We remark here that we use a U-Net in both the branch and t-branch networks.

1. Lift the input functions $v(t_i, \cdot)$ and the full solution history $\mathrm{U}_{\mathrm{hist}}(t) = \{u(t-(l-1)\Delta t, \mathrm{x})\}_{l=1}^{L}$ to separate latent representation spaces using two independent linear encoders P_b and P_{tb}:

$$\begin{aligned} h_b(v) &= P_b(v(t_i, \cdot)) \in \mathbb{R}^p, \\ h_{tb}(\mathrm{U}_{\mathrm{hist}}(t)) &= P_{tb}(\mathrm{U}_{\mathrm{hist}}(t)) \in \mathbb{R}^p. \end{aligned} \tag{8}$$

The branch encoder processes the input function v, while the t-branch encoder processes the full temporal history of the solution $\mathrm{U}_{\mathrm{hist}}$. Each encoder maps its input into a latent space of dimension p, but these mappings are learned independently.

2. Process the aligned latent representations through U-Net architectures in both the branch and t-branch. To handle variability in spatial resolution, the latent feature maps $h_b(v)$ and $h_{tb}(\mathrm{U}_{\mathrm{hist}}(t))$ are first passed through 2D adaptive average pooling to produce a fixed spatial resolution. The pooled features are then processed through separate U-Net blocks, followed by bilinear upsampling to restore the feature maps to the original spatial resolution $H \times W$ of the input function or data grid:

$$\begin{aligned} q_b &= \mathrm{AdaptiveAvgPool2d}(h_b(v)) \in \mathbb{R}^{p\times S\times S}, \\ q_{tb} &= \mathrm{AdaptiveAvgPool2d}(h_{tb}(\mathrm{U}_{\mathrm{hist}}(t))) \in \mathbb{R}^{p\times S\times S}, \\ U_b &= \text{U-Net}_b(q_b) \in \mathbb{R}^{p\times S\times S}, \\ U_{tb} &= \text{U-Net}_{tb}(q_{tb}) \in \mathbb{R}^{p\times S\times S}, \\ \tilde{U}_b &= \mathrm{Upsample}(U_b) \in \mathbb{R}^{p\times H\times W}, \\ \tilde{U}_{tb} &= \mathrm{Upsample}(U_{tb}) \in \mathbb{R}^{p\times H\times W}. \end{aligned} \tag{9}$$

We found that it is often beneficial to set the pooling resolution $S \times S$ to the lowest available input resolution, as this reduces memory usage and improves training efficiency. In practice, it may not be necessary to explicitly define or tune S, as the resolution can be inferred from the data or heuristically selected. This combination of adaptive pooling, U-Net, and upsampling allows the network to generalize across inputs of varying spatial resolutions while maintaining high representational capacity. It decouples the input resolution from the network architecture, enabling the model to operate on arbitrary grid sizes during inference without retraining.

3. Encode the spatio-temporal coordinates using the trunk network. The trunk takes as input the coordinates (t, x), where $t \in \mathbb{R}$ denotes the time, and $\mathrm{x} \in \mathbb{R}^m$ denotes the spatial location. These coordinates are passed through a fully connected feedforward neural network f_θ, typically with nonlinear activation functions (e.g., hyperbolic tangent), to produce a feature vector in the latent space $\mathbb{R}^p$:

$$T_i(\mathrm{x}, t) = f_\theta(\mathrm{x}, t) \in \mathbb{R}^{p\times H\times W}. \tag{10}$$

The trunk features $t_i(\mathrm{x}, t)$ serve as the coordinate-dependent embedding that interacts with the outputs of the branch and t-branch through an element-wise (Hadamard) product.

4. The full TNO output can be written in operator form as:

$$\widehat{\mathrm{U}}_{\mathrm{fut}}(t)(\mathrm{x}) = \mathcal{G}_\theta^{L\to K}(\mathrm{U}_{\mathrm{hist}}(t))(\mathrm{x}) = G\left(\tilde{U}_b(\mathrm{x}, t) \odot \tilde{U}_{tb}(\mathrm{x}, t) \odot T_i(\mathrm{x}, t)\right), \tag{11}$$

where $\mathcal{G}_\theta^{L\to K}$ denotes the learned operator mapping a temporal history of L past solution states to a bundled prediction of K future time steps, $\odot$ is the Hadamard product along the latent feature dimension p, and $G : \mathbb{R}^p \to \mathbb{R}^K$ is a shared MLP decoder applied pointwise over the spatial domain. The output $\mathcal{G}_\theta^{L\to K}(\mathrm{U}_{\mathrm{hist}}(t)) \in \mathbb{R}^{K\times H\times W}$ represents a temporally bundled solution over the full spatial grid.

The TNO architecture is implemented in PyTorch and trained on an NVIDIA Tesla V100 GPU. The ADAM optimizer is used to minimize the mean square error (MSE) loss between the predicted and ground truth solutions. The same U-Net architecture is employed for both the branch and temporal branch across all examples. Additional architectural details are provided in Supplementary S1, and summarized in Supplementary Table S1.

Results

Weather forecast for European air temperature

The E-OBS (European Observations) dataset[31] is a high-resolution observational climate dataset designed for analyzing climate variability and long-term trends across Europe. It covers a broad geographical region (approximately $25°\mathrm{N} - 71.5°\mathrm{N} \times 25°\mathrm{W} - 45°\mathrm{E}$) and includes several decades of daily mean temperature measurements. A key feature of E-OBS is its high spatial resolution, available at $0.1°$ and $0.25°$, corresponding to regular grids of size 705×465 and 201×464, respectively. This fine-grained spatial coverage enables detailed regional weather and climate analysis. Unlike datasets generated or constrained by numerical models, E-OBS is constructed directly from European weather station observations. As a result, the dataset presents unique challenges for learning-based models: spatial gaps, temporal discontinuities, and evolving coverage patterns introduce inconsistencies that the TNO must learn to handle during training and inference.

For this experiment, we train the TNO with an input bundle size of $L = 1$ and an output bundle size of $K = 4$, using daily mean temperature and pressure fields as input variables. The training set spans 9,000 days from 1997 to 2022. Validation is performed on 360 days from January 1, 2022, to December 26, 2022, and testing is conducted on 360 days from January 1, 2023, to December 26, 2023. All training and testing are conducted on the lower-resolution version of the dataset ($0.25°$, corresponding to a 201×464 grid). To evaluate the TNO's resolution invariance, an additional high-resolution test set ($0.1°$, 705×465) is constructed for the period December 26, 2022, to December 31, 2023. This setup allows us to assess whether the TNO can generalize to unseen spatial resolutions without retraining.

The training, validation, and testing datasets are standardized using z-score normalization computed from per-pixel statistics of the training set. Each training sequence is structured into bundles of nine time steps: the first snapshot serves as the initial condition, followed by two rollout windows of size four for autoregressive supervision. More details on the training and hyperparameters are provided in Supplementary S2. This dataset highlights several key capabilities of the TNO architecture:

- *Forecasting under real-world observational noise and gaps* the TNO effectively learns from gridded observational data with missing values and temporal-spatial inconsistencies, showcasing robustness to incomplete or imperfect datasets.
- *Multi-day sequence prediction with minimal historical context* with an input sequence length of $L = 1$, the TNO successfully generates forecasts over a multi-day horizon ($K = 4$), demonstrating its ability to model short-term atmospheric dynamics with limited history.
- *Generalization to unseen high-resolution spatial grids* the TNO accurately forecasts temperature fields on a higher-resolution grid ($0.1°$) at test time, despite being trained exclusively on lower-resolution data ($0.25°$), indicating resolution-invariant generalization.

Results of weather forecast for European air temperature

To evaluate the TNO performance for regional weather forecasting, we utilize a blind test dataset from the year 2023. During testing, the TNO is initialized with initial conditions comprising daily mean air temperature, pressure, and the spatial grid. The model then performs two autoregressive rollouts of size $2 \times K$, predicting air temperatures across successive eight-day intervals. After each prediction cycle, a new initial condition is drawn from the ground truth to initialize the next rollout window. This approach simulates a realistic deployment setting, where updated observational data is periodically supplied and the model forecasts the near-future temperature evolution.

The mean absolute error (MAE) and root mean square error (RMSE) of the predicted temperatures per time snapshot is shown in 4, with an overall MAE of $2.68°\mathrm{C}$ across all snapshots. The results indicate that the TNO does not incur significant error accumulation over successive rollouts. Two qualitative examples of predicted air temperatures for May 06, 2023 and November 12, 2023 are shown in Figures 2a and 3a, respectively, both of which demonstrate excellent agreement with the ground truth. However, it seems that the TNO slightly struggles with capturing temperature extremes, which can be observed in the northern most parts of Europe (Fig. 2a, where some smoothing also accrues. Two additional trajectories at the lower $0.25°$ resolution are shown in Supplementary Figs. S1a and S2a under Supplementary S3. These findings underscore the TNO's effectiveness in temporal extrapolation and its ability to maintain stable prediction quality over extended forecasting horizons.

To assess the TNO's resolution invariance, we employ a second testing dataset for the same period (2023), rendered at a higher resolution of $0.1°$. Remarkably, the TNO achieves an overall MAE of $2.83°\mathrm{C}$ on this dataset–without any additional training or fine-tuning–highlighting the model's capacity to generalize across spatial resolutions. The per-snapshot MAE and RMSE for the high-resolution test set is shown in Fig. 4, and corresponding predicted temperature fields for May 06, 2023 and November 12, 2023 are visualized in Figures 2a and 3a, respectively. These results further demonstrate the robustness and versatility of the TNO in handling variations in spatial discretization while maintaining high predictive accuracy. Despite the excellent agreement between the low and high-resolution tests, testing with super resolution data only seems to exacerbate the TNO struggle with temperature extremes, as can be seen in Northern parts of Europe and the Middle-East in Fig. 2b. The smoothing effect is less prevalent in Fig. 3 where the temperature gradient is lower. Two additional trajectories at the finer $0.1°$ resolution are shown in Supplementary Figs. S1b and S2b. To improve the TNO performance in the super-resolution task, we can opt for fine-tuning the TNO with high-resolution data during inference, or train the TNO on both low and high-resolution data simultaneously. Both of these approaches can be implemented easily. However, this is beyond the scope of this paper which focuses on the raw performance of the TNO.

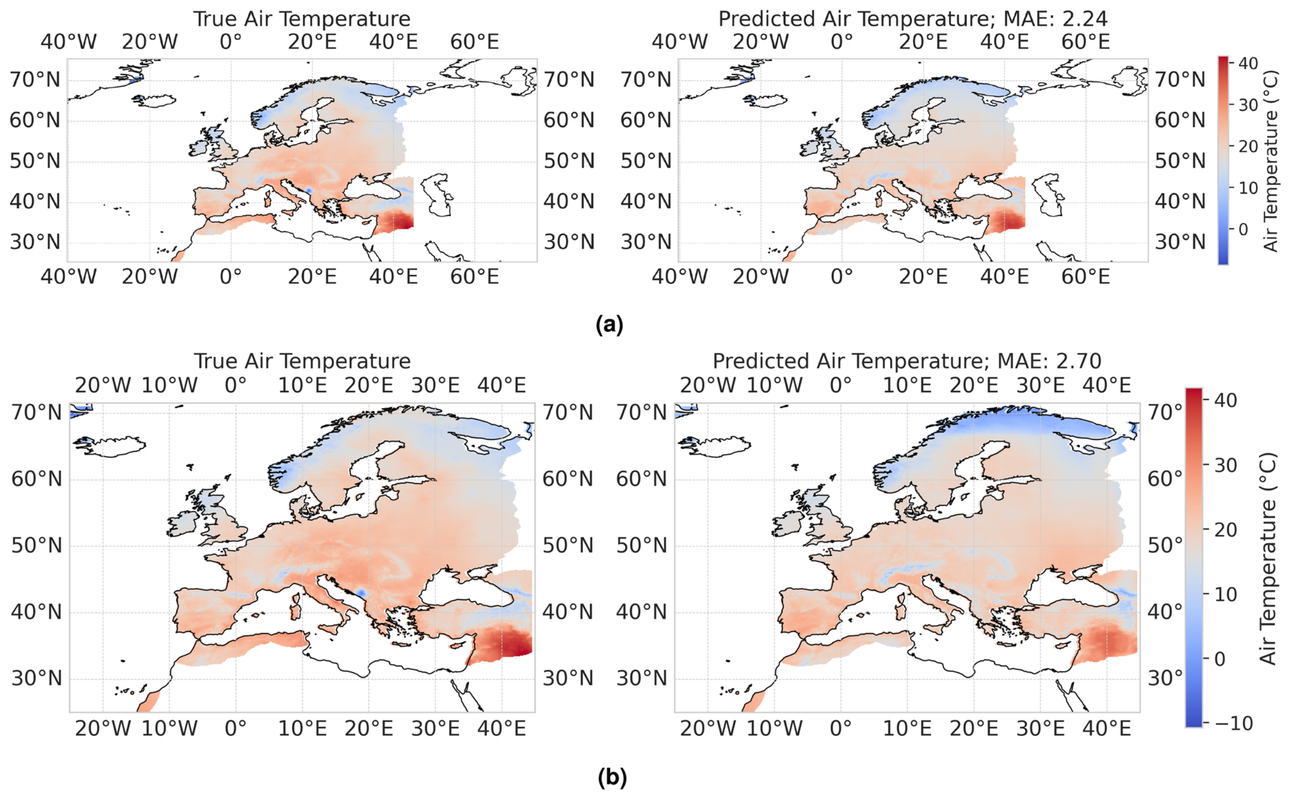


Fig. 2. Qualitative testing performance of the TNO - Summer 2023.

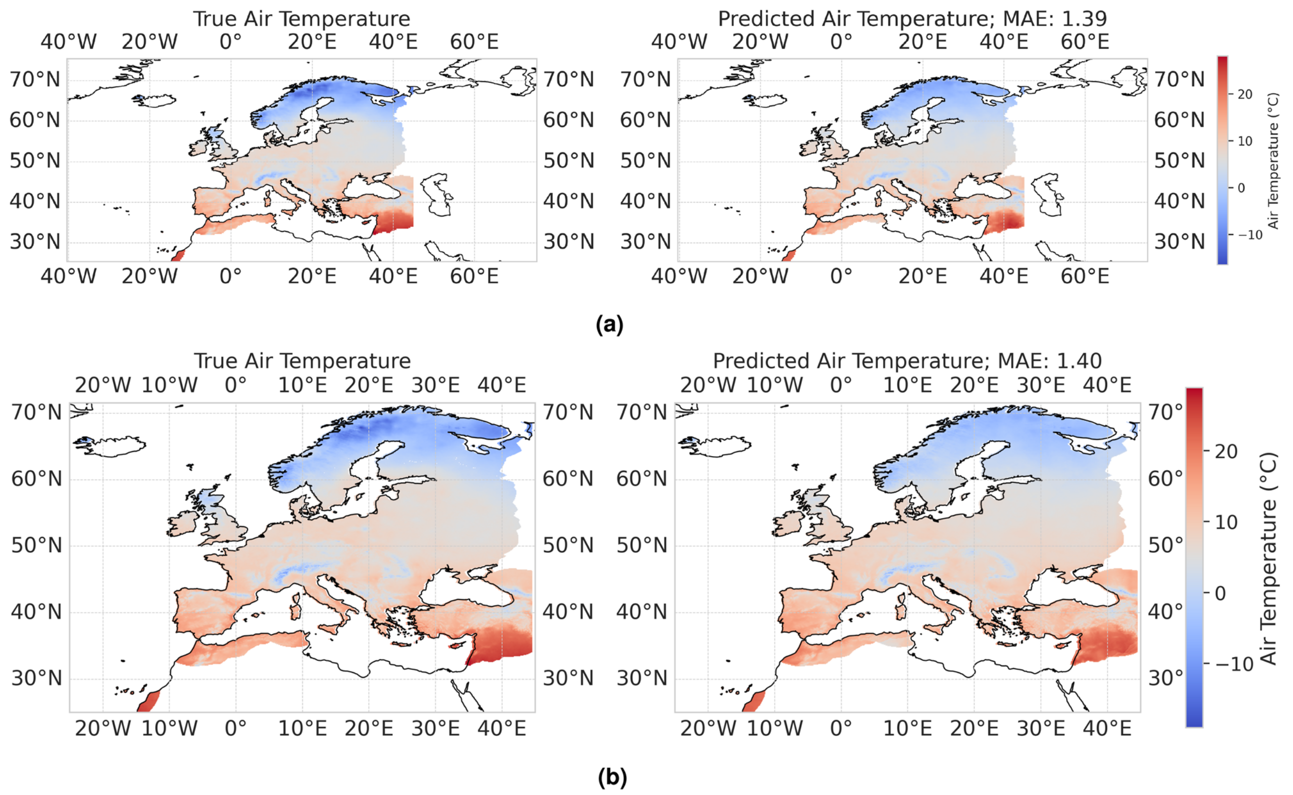


Fig. 3. Qualitative testing performance of the TNO - Winter 2023.

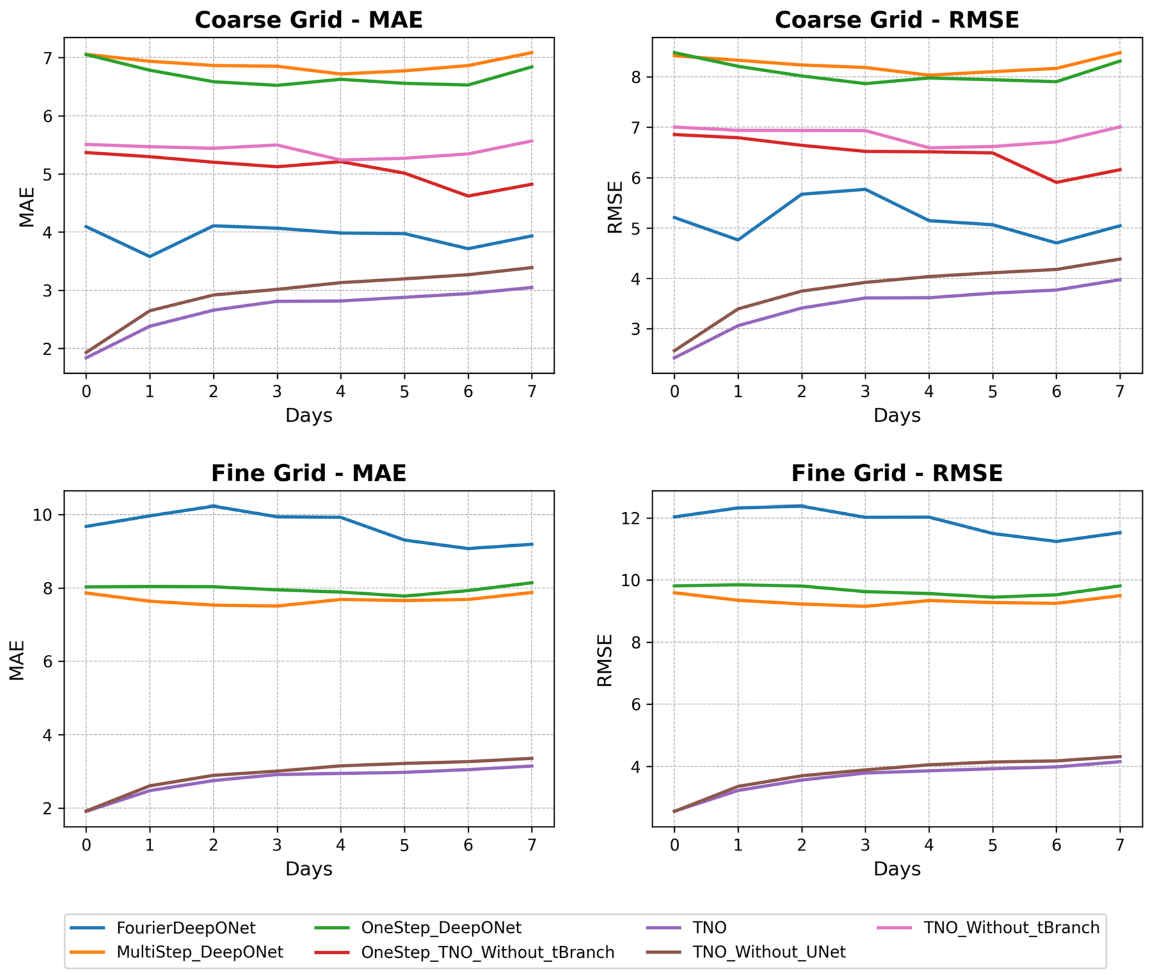


Fig. 4. MAE (left) and RMSE (right) on the coarse (top) and fine (bottom) grids. "Multistep" denotes temporal bundling ($K = 4$), "Onestep" predicts one step at a time.

Ablation study

To asses the contribution of the individual components in our TNO, we evaluate three ablated variants alongside the full TNO on the 2023 coarse-grid test set (0.25°):

- *TNO without t-Branch* the temporal branch is removed, so predictions rely only on the branch (U-Net) and trunk.
- *TNO without U-Net* both the branch and temporal branch U-Net blocks are replaced by MLPs.
- *One step TNO without t-Branch* combines the previous two ablations (no t-branch and $K = 1$).
- *One step TNO* the full TNO architecture (t-branch, U-Net) but predicting one step at a time ($K = 1$), i.e. no temporal bundling.
- *Full TNO* the complete architecture as presented, with t-branch, U-Net spatial encoders, and temporal bundling ($L = 1, K = 4$).

Figure 4 presents hlper time snapshot RMSE and MAE for these variants on both the coarse (top row) and fine (bottom row) grids. On the coarse grid, removing the temporal branch (`TNO without tBranch`) roughly doubles both RMSE and MAE and also eliminates the model's grid-invariance capability. Replacing the U-Net with MLPs (`TNO without UNet`) leads to a significant increase in spatial error, underscoring the importance of deep spatial feature extraction. The combined ablation (`OneStep TNO without tBranch`) performs comparably to `TNO without tBranch`, indicating that temporal bundling alone cannot compensate for the absence of the t-branch. Overall, the Full TNO achieves the lowest RMSE and MAE across all forecast snapshots, validating the complementary roles of the U-Net, temporal branch, and temporal bundling. It is worth noting, however, that the performance improvement of the U-Net compared to the FNN remains modest. This is likely due to the low number of input channels (temperature and surface pressure), which limits the need for deep multiscale spatial encoding. Both input fields are relatively smooth and correlated, allowing even the simpler FNN-based variant to achieve competitive performance. Note that the ablation experiments were repeated with multiple random seeds. Since the results showed low variance across runs, we report outcomes from a randomly selected seed.

Benchmark comparisons

We compare the full TNO against several state-of-the-art neural operators on the 2023 test sets at both coarse (0.25°) and fine (0.1°) resolutions. The baselines are:

- *DeepONet (one-step)* Vanilla DeepONet trained autoregressively to predict one day ahead ($K = 1$), without temporal bundling.
- *DeepONet (multi-step)* Vanilla DeepONet with temporal bundling ($L = 1, K = 4$), predicting four days in each forward pass.
- *Fourier-DeepONet*[32]: A Fourier Neural Operator variant adapted for temporal extrapolation with the same bundling ($L = 1, K = 4$).
- *TNO (full)* The Temporal Neural Operator with t-branch, U-Net spatial encoders, and temporal bundling ($L = 1, K = 4$).

Figure 4 (top row) reports per-snapshot RMSE and MAE for these methods on the coarse grid, while the bottom row reports similar results for the fine grid predictions. On the coarse grid, the Full TNO consistently achieves the lowest MAE and RMSE at all lead times, outperforming both OneStep and Multi step DeepONets and the Fourier-DeepONet by a wide margin. When evaluated at the higher 0.1° resolution without any retraining, the Full TNO's error increases only marginally, whereas Fourier-DeepONet and the DeepONet variants suffer substantial degradation. These results demonstrate the TNO's superior accuracy, stability over long horizons, and robustness to changes in spatial resolution.

Climate modeling for global air temperature

Modeling the Earth's climate involves simulating complex, nonlinear, and multi-scale dynamics across interacting systems such as the atmosphere, oceans, land, and ice. Traditional approaches, like General Circulation Models (GCMs)[33], solve time-dependent partial differential equations (PDEs) governing fluid flow, heat transfer, and radiation. They are computationally expensive and rely on parameterizations for unresolved processes fraught with uncertainty. Recent machine learning advancements, including foundation models[34,35], offer data-driven alternatives but still require vast datasets and compute budgets[36]. In contrast, the TNO presented here is designed as a lightweight, computationally efficient framework for solving time-dependent PDEs directly from spatio-temporal data, making it a practical tool for scientific modeling in resource-constrained settings.

Our objective in this experiment is to leverage the TNO to efficiently and accurately model the spatio-temporal evolution of global air temperature, while reducing computational costs and enabling forecasts across all atmospheric pressure levels. This experiment demonstrates three core capabilities of the TNO:

- *Long-term temporal extrapolation* The TNO accurately forecasts global air temperature fields over a five-year horizon, demonstrating strong extrapolation performance beyond the temporal range seen during training.
- *Vertical generalization across pressure levels* The model successfully interpolates and extrapolates temperature fields at atmospheric pressure levels not included in the training set, highlighting its ability to generalize across the vertical dimension.
- *Computationally efficient 3D modeling via 2D operations* By treating the 3D spatio-temporal climate data as a collection of 2D slices conditioned on pressure levels, the TNO leverages 2D convolutional architectures to efficiently learn in a high-dimensional setting.
- *Unified multi-level forecasting* Unlike models trained on individual pressure levels, the TNO uses a single architecture to forecast temperature fields across all levels simultaneously, this reduces model complexity and improves generalization.

To evaluate these capabilities, we use the NCEP/NCAR Reanalysis 1 dataset[37], which provides daily mean air temperature fields on a global 144×72 spatial grid (2.5° resolution) across 16 pressure levels, spanning from 1948 to the present. For this experiment, we use data from January 1, 2010 to December 31, 2015 for training, January 1, 2016 to December 31, 2018 for validation, and January 1, 2019 to December 31, 2023 for testing.

The temporal branch receives a history of air temperature fields $\mathrm{T}_{\text{hist}}(t)$, while the branch network is conditioned on the atmospheric pressure level P. This formulation allows the TNO to both interpolate and extrapolate air temperature across pressure levels, while treating the 3D temperature field as a series of 2D slices. To demonstrate this, we train the TNO using data from 12 out of the 16 available pressure levels: $\{1000, 925, 850, 600, 300, 250, 200, 150, 70, 50, 30, 20\}$ mb, holding out 4 pressure levels for evaluation.

The dataset was standardized using z-score normalization computed from per-pixel statistics of the training set. We define the temperature field at time t and pressure level index p as $\mathrm{T}_p(x, y, t)$, where (x, y) denotes spatial coordinates on the global grid. For each training example, we construct an input tensor $\mathrm{T}_{\text{hist},p} \in \mathbb{R}^{H\times W\times L}$, representing the temperature field at pressure level p over L past time steps. The corresponding target is a tensor $\hat{\mathrm{T}}_{\text{fut},p} \in \mathbb{R}^{H\times W\times K}$, representing predicted temperatures at the same level over the next K time steps. In this experiment, we set both the input and output sequence lengths to one year: $L = K = 365$. During training, the TNO is provided with five years of daily temperature data, spanning from January 1, 2010 to December 31, 2015. The model is trained across 12 distinct pressure levels from the available dataset, with each level treated as a separate input condition to the branch network. To improve training efficiency and promote cross-level generalization, pressure levels are batched in groups of four. That is, during each training step, the TNO receives a batch of four temperature tensors $\mathrm{T}_{\text{hist},p}$ corresponding to four distinct pressure levels, and jointly predicts the corresponding future sequences $\hat{\mathrm{T}}_{\text{fut},p}$. For each year and pressure level p, the temporal branch receives an input tensor $\mathrm{T}_{\text{hist},p} \in \mathbb{R}^{144\times 72\times 365}$ from year t, and the TNO predicts $\hat{\mathrm{T}}_{\text{fut},p}$ for year $t + 1$, using the corresponding ground truth as supervision. This process is repeated sequentially across all training years and level batches. Finally, we use teacher forcing during training, where ground truth sequences are used to guide future prediction and accelerate convergence. During inference, the TNO generates multi-step forecasts by recursively using its own predictions as input. Additional training details and hyperparameter configurations are provided in Supplementary S4.

Results of global air temperature forecast

To evaluate the performance of the TNO, we use a blind test set comprising five years of daily mean air temperature data, spanning all 16 atmospheric pressure levels $\{1000, 925, 850, 700, 600, 500, 400, 300, 250, 200, 150, 100, 70, 50, 30, 20\}$ mb, from January 1, 2019 to December 31, 2023. This setup is designed to test the TNO's ability to (i) extrapolate temperature fields forward in time, and (ii) interpolate and extrapolate across pressure levels not included in training. The evaluation metric used is the relative L_2 error, computed across all spatial grid points and time steps.

This task represents a challenging generalization scenario in which the TNO must forecast into an unseen temporal domain and make predictions at pressure levels it has not encountered during training. As shown in Fig. 5d, the TNO achieves strong performance in both respects, with a mean relative L_2 error of 0.016. The error is generally higher at higher-altitude (lower-pressure) levels, likely due to data imbalance across pressure levels in the training set.

Additionally, at 1000 mb–the lowest atmospheric level–the TNO achieves accuracy on par with the Ditto transformer[38], which was trained specifically for that level and for the same extrapolation period. However, unlike Ditto, the TNO generalizes across all levels with a single unified model. Figure 5a and 5b show a visual comparison between the TNO's predictions and the ground truth air temperature fields at 1000 mb for selected dates from the testing period (2020–2023), while Fig. 5c show a visual comparison between the TNO's predictions and the ground truth air temperature fields at 700 mb. Additional qualitative and quantitative results are provided in Supplementary S5, where two more examples of air temperature fields at 1000 mb are shown in Supplementary Fig. S3, and three additional examples of air temperature field at various other pressure levels are shown in Supplementary Fig. S4.

Geologic carbon sequestration

In geological carbon sequestration (GCS), captured CO_2 is injected into deep subsurface formations for long-term storage, thereby reducing atmospheric emissions[39]. Accurate predictions of CO_2 plume migration and pressure evolution are essential for ensuring containment, optimizing injection strategies, and satisfying regulatory monitoring requirements. Modeling GCS processes involve complex, coupled multiphase flow and transport governing PDEs. Simulations are computationally demanding, as they must span decades and account for site-specific heterogeneities[40–42].

In this final example, we employ the TNO to model CO_2 plume migration and pressure buildup in an underground geological storage site. We utilize the dataset published in[9] to evaluate the TNO's performance for GCS problems. The same dataset was used in our earlier work[10] to train various neural operator architectures; however, none of those architectures, including[9], were able to extrapolate well beyond the temporal training horizon of the dataset. In this work, we evaluate the TNO's ability to (i) extrapolate plume migration and pressure buildup over extended timescales, and (ii) generalize across varying geological conditions, embedding both the elliptic and hyperbolic PDE dynamics in its learned operator.

Training and inference for GCS

We base our experiments on the benchmark GCS dataset from[9], which comprises 5,500 realizations of input–output mappings for gas saturation S_g and pressure buildup ΔP. Each realization contains solutions at 24 successively coarsening time snapshots $\{1\,\text{day}, 2\,\text{days}, 4\,\ldots, 323\,\text{days}, 1.3\,\text{years}, \ldots, 21.1\,\text{years}, 30\,\text{years}\}$. The original split (9:1:1) allocates 4,500 realizations for training, 500 for validation, and 500 for testing. To evaluate long-term extrapolation, we restrict both training and validation to the first 16 snapshots $\{1\ \text{day}, \ldots, 1.8\ \text{years}\}$, reserving the final 8 snapshots $\{2.6\ \text{years}, \ldots, 30\ \text{years}\}$ for blind testing of temporal extrapolation and generalization.

Each input realization consists of four spatial fields–horizontal permeability k_x, vertical permeability k_y, porosity ϕ, and perforation height h_{perf} (each $\in \mathbb{R}^{H\times W}$)–and five scalars: initial reservoir pressure P_{init}, injection rate Q, temperature T, capillary scaling λ, and irreducible water saturation S_{wi}. The outputs are the saturation field $\mathrm{S}_g \in \mathbb{R}^{H\times W}$ and pressure buildup field $\Delta\mathrm{P} \in \mathbb{R}^{H\times W}$. We direct the readers to the original paper[9] for more details on the reservoir description, the numerical simulation, the generation of the field maps and all other sampling techniques for the inputs.

Our objective in this experiment is to leverage the TNO to efficiently and accurately model the spatio-temporal evolution of CO_2 plume migration and pressure buildup in subsurface formations, while reducing computational cost and enabling long-term forecasts under heterogeneous geological conditions. This experiment demonstrates three core capabilities of the TNO:

- *Long-term temporal extrapolation* Evaluate forecasting accuracy for saturation and pressure fields over multi-decadal horizons beyond the training time window ($t > 1.8$ years).
- *Generalization to unseen geological conditions* Test robustness to new input functions including diverse permeability, porosity fields, and well-configuration not seen during training.
- *Coupled multiphysics modeling* Examine the TNO's capacity to concurrently capture elliptic pressure dynamics and hyperbolic saturation transport within a unified operator framework.

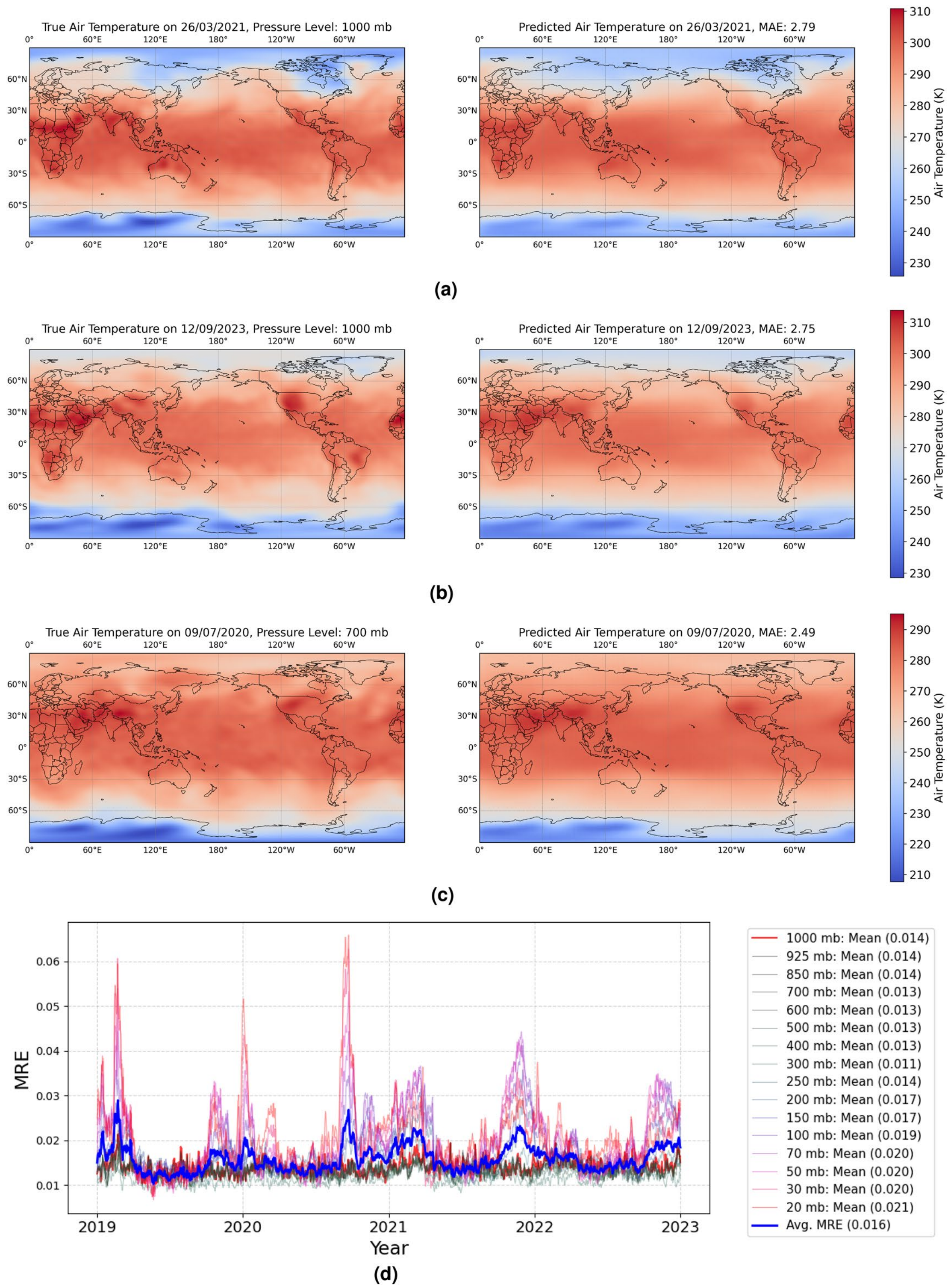
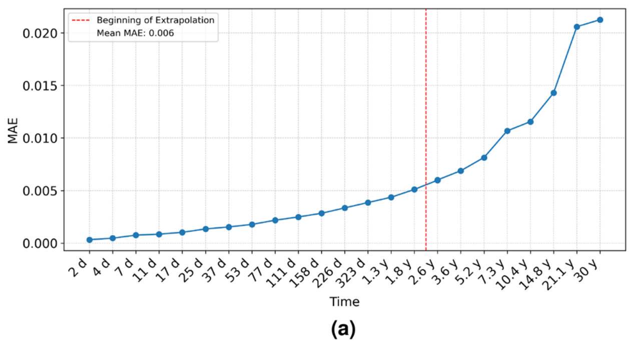


Fig. 5. Testing performance of the TNO for Global Temperature Forecast. Note that the initial condition of year 2019 are excluded from the error calculation.

Following the TNO architecture (Fig. 1), the four input fields and five scalars are encoded by the branch network; the initial condition $\mathrm{S}_g(x, y, t_0),\ \Delta\mathrm{P}(x, y, t_0)$ is processed by the t-branch, and the temporal grid is handled by the trunk. We set the history length and forecast horizon to $L = 1$ and $K = 3$, respectively, during training and testing. More information about training and hyperparameters can be found in Supplementary S6.



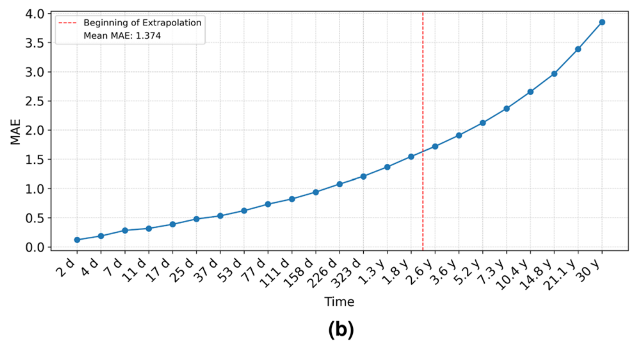


Fig. 6. Evaluation metrics of simultaneous generalization and extrapolation to unseen realizations and time snapshots for all CO_2 saturation and pressure buildup (dP) time snapshots. Extrapolation begins after 1.8 years. Results are averaged over 500 realizations.

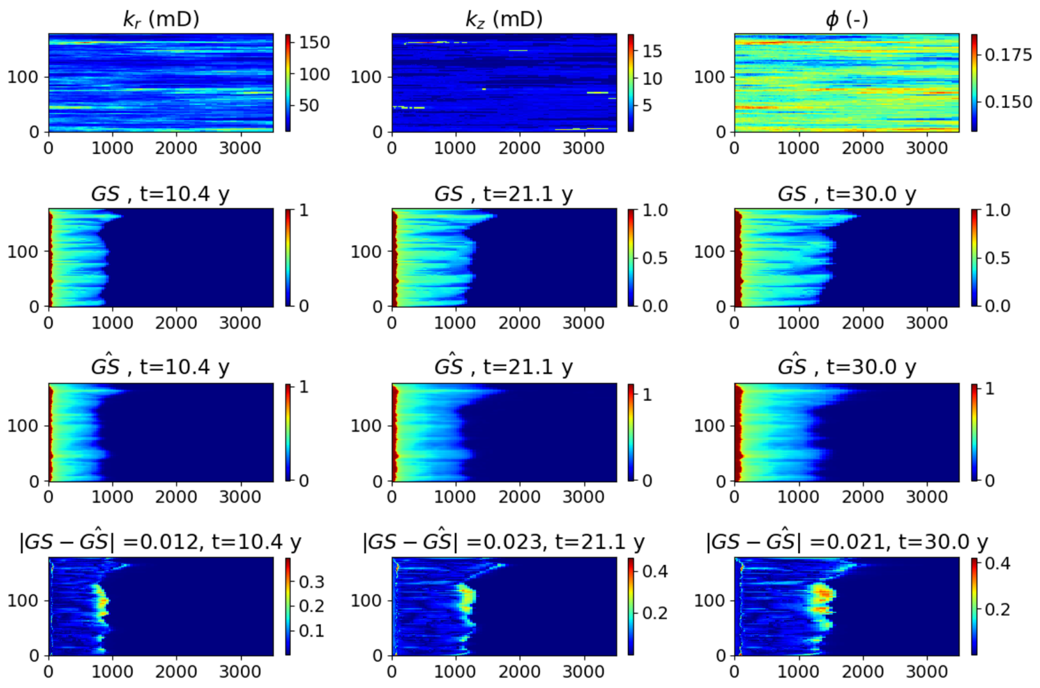


Fig. 7. An example of CO_2 saturation performance in simultaneous extrapolation and generalization. *Row 1*: Input reservoir properties: radial permeability (k_r), vertical permeability (k_z), and porosity (ϕ). *Row 2*: Ground truth saturation field. *Row 3*: TNO predicted saturation. *Row 4*: Absolute error.

Results of geologic carbon sequestration

To assess the performance of the TNO in simultaneous generalization and temporal extrapolation tasks for both CO_2 saturation and pressure buildup, we utilize the testing dataset which consists of 500 previously unseen realizations. Importantly, time snapshots beyond 1.8 years were excluded during training, which allows us to classify performance on any time snapshot within the training time range as generalization, and performance beyond 1.8 years as simultaneous generalization and temporal extrapolation. The mean absolute error serves as the primary metric for quantifying TNO performance. For CO_2 saturation specifically, error calculations are restricted to the CO_2 saturation plume region, as saturation values outside this area remain zero. Figure 6 shows the MAE for the saturation 6a and for the pressure buildup 6b. Both figures clearly demonstrate that the TNO effectively generalizes and extrapolates to unseen realizations of gas saturation and pressure buildup, as well as to time snapshots not included in the training data. There is, however, a slight decline in solution quality towards the end of the temporal domain, which is to be expected since we are further out in the extrapolation. Figures 7 and 8 show an example of CO_2 saturation plume and CO_2 pressure buildup towards the end of the extrapolation period, respectively. Two more examples of CO_2 saturation plume with different input parameters are shown in Supplementary Figs. S5 and S6, and two additional pressure buildup examples with different input parameters are shown in Supplementary Figs. S7 and S8 under Supplementary S7).

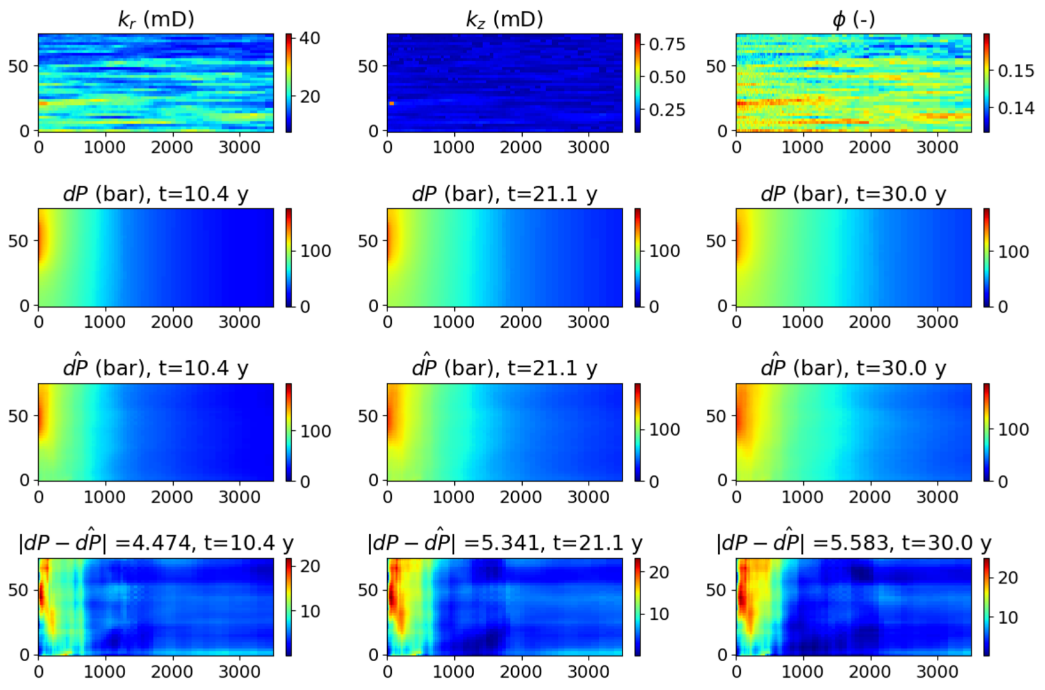


Fig. 8. An example of pressure buildup (dP) performance in simultaneous extrapolation and generalization. *Row 1*: Input reservoir properties: radial permeability (k_r), vertical permeability (k_z), and porosity (ϕ). *Row 2*: Ground truth pressure buildup field. *Row 3*: TNO predicted pressure buildup. *Row 4*: Absolute error.

Dataset	Trainable parameters (Million)	GPU memory (GB)	Time/Epoch (s)
Global temperature forecast	0.12	6.4	39.0
European temperature forecast	39.4	12.2	3.63
Carbon sequestration - GS	2.66	4.3	211.0
Carbon sequestration - dP	7.41	7.5	447.0

Table 1. Performance summary of the TNO.

Discussion

This paper introduces the Temporal Neural Operator (TNO) for reliable generalization and temporal extrapolation of parametric time-dependent PDEs. The TNO's performance was evaluated across three distinct problem domains: forecasting global air temperature across all atmospheric levels for climate modeling, predicting regional air temperatures for Europe and adjacent regions, and modeling flow and transport in geologic carbon sequestration, involving the tracking of CO_2 saturation plume and pressure buildup. Despite the diverse nature of these problems and the unique challenges posed by their respective datasets, the TNO consistently demonstrated remarkable performance. These results further highlight our model's versatility in handling complex spatio-temporal dynamics across various scientific applications.

The TNO addresses several longstanding challenges in Scientific Machine Learning (SciML). Namely, simultaneous long-term extrapolation and generalization, grid invariance, computational efficiency and low memory foot-print. The TNO is a lightweight, efficient architecture that can operate seamlessly on a single off-the-shelf GPU. For example, in the climate modeling problem, the TNO utilizes only 7.5 GB of GPU memory (see Table 1 for a complete summary on TNO performance). The TNO requires about 3.63 seconds per epoch (batch size of 4) for a total training time of only 7.2 minutes. Moreover, we take advantage of the generalization properties of the TNO to interpolate between atmospheric levels, which reduces the complexity of the 3D problem to a 2D learnable problem without any loss of information. By feeding the spatio-temporal grid to the trunk of the TNO, we learn the temporal dimension through a feed-forward neural network, further reducing the dimensionality of the problem. Importantly, the dimensionality reduction techniques applied in this example are confined to the learning process and do not alter the problem formulation itself. This level of flexibility and adaptability underscores the TNO's robust learning capacity and efficiency.

In the weather forecasting problem, the TNO grid invariance capabilities were demonstrated. The TNO is trained on air temperature data of resolution 0.25° taken from the E-OBS (European Observations) dataset[31]. TNO is then tested on unseen data with a resolution of 0.1°. The accuracy in the temporal extrapolation task is quite remarkable given that no additional training or fine-tuning was needed, despite the inherent challenges in the dataset due to missing sensor data in some daily measurements and the changing areal coverage of the dataset. Moreover, while the resolution generalization study in the E-OBS dataset focuses on horizontal spatial refinement, we note that extending this to homogeneously refined 3D grids (e.g., including vertical atmospheric levels) may introduce additional challenges due to the increased spatial variability and anisotropy. We expect the TNO to remain effective when the initial conditions are defined on the target resolution, but further empirical validation is needed to fully characterize performance under vertical refinement.

In the geological carbon sequestration problem, the TNO was trained on data representing the first two years of CO_2 storage, corresponding to 16 time steps. Despite the limited temporal scope of the training data and the inherent complexity of the problem, the TNO demonstrated exceptional capabilities. It effectively generalized to new instances of the problem, which involved nine varying PDE parameters, while simultaneously extrapolating temporal dynamics for up to 30 years. This performance highlights the TNO's robustness and reliability, even in scenarios with limited data.

Data availability

The raw data required to reproduce the European temperature forecasts is available from the EU-FP6 project UERRA https://www.uerra.eu and the Copernicus Climate Change Service, and the data providers in the ECA&D project https://www.ecad.eu. The raw data required to reproduce the global temperature forecasts are available on the NOAA Physical Sciences Laboratory website (https://psl.noaa.gov/data/gridded/index.html). The raw data required to reproduce the geologic carbon sequestration dataset is available on https://drive.google.com/drive/folders/1fZQfMn_vsjKUXAfRV0q_gswtl8JEkVGo?usp=sharing and in the U-FNO paper4. For more information about the data used in the study please contact the corresponding author.

Received: 28 April 2025; Accepted: 20 August 2025
Published online: 25 September 2025

References

1. Reichstein, M. et al. Deep learning and process understanding for data-driven earth system science. *Nature* **566**, 195–204 (2019).
2. Carrassi, A., Bocquet, M., Bertino, L. & Evensen, G. Data assimilation in the geosciences: An overview of methods, issues, and perspectives. *Wiley Interdiscip. Rev. Clim. Chang.* **9**, e535 (2018).
3. Raissi, M., Perdikaris, P. & Karniadakis, G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **378**, 686–707. https://doi.org/10.1016/j.jcp.2018.10.045 (2019).
4. Krishnapriyan, A., Gholami, A., Zhe, S., Kirby, R. & Mahoney, M. W. Characterizing possible failure modes in physics-informed neural networks. *Adv. Neural. Inf. Process. Syst.* **34**, 26548–26560 (2021).
5. Wang, S., Yu, X. & Perdikaris, P. When and why pinns fail to train: A neural tangent kernel perspective. *J. Comput. Phys.* **449**, 110768 (2022).
6. Zhao, C., Zhang, F., Lou, W., Wang, X. & Yang, J. A comprehensive review of advances in physics-informed neural networks and their applications in complex fluid dynamics. *Phys. Fluids* **36**, 101301 (2024).
7. Lu, L., Jin, P., Pang, G., Zhang, Z. & Karniadakis, G. E. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nat. Mach. Intell.* **3**, 218–229. https://doi.org/10.1038/s42256-021-00302-5 (2021).
8. Li, Z. et al. Fourier neural operator for parametric partial differential equations. arXiv:2010.08895. (2021).
9. Wen, G., Li, Z., Azizzadenesheli, K., Anandkumar, A. & Benson, S. M. U-FNO-An enhanced Fourier neural operator-based deep-learning model for multiphase flow. *Adv. Water Resour.* **163**, 104180. https://doi.org/10.1016/j.advwatres.2022.104180 (2022).
10. Diab, W. & Al Kobaisi, M. U-DeepONet: U-Net enhanced deep operator network for geologic carbon sequestration. Preprint at arXiv:2311.15288https://doi.org/10.48550/arXiv.2311.15288 (2023).
11. Jiang, Z., Zhu, M. & Lu, L. Fourier-mionet: Fourier-enhanced multiple-input neural operators for multiphase modeling of geological carbon sequestration. *Reliab. Eng. Syst. Saf.* **251**, 110392 (2024).
12. Lee, J. E. et al. Efficient and generalizable nested Fourier-deeponet for three-dimensional geological carbon sequestration. *Eng. Appl. Comput. Fluid Mech.* **18**, 2435457 (2024).
13. Zappala, E. et al. Learning integral operators via neural integral equations. *Nat. Mach. Intell.* **6**, 1046–1062 (2024).
14. Kovachki, N. et al. Neural operator: Learning maps between function spaces with applications to pdes. *J. Mach. Learn. Res.* **24**, 1–97 (2023).
15. Bonev, B. et al. Spherical Fourier neural operators: Learning stable dynamics on the sphere. In *International conference on machine learning*, 2806–2823 (PMLR, 2023).
16. Nayak, D. & Goswami, S. Ti-deeponet: Learnable time integration for stable long-term extrapolation. Preprint at arXiv:2505.17341 (2025).
17. Khurjekar, I., Saha, I., Graham-Brady, L. & Goswami, S. Enhanced accuracy through ensembling of randomly initialized auto-regressive models for time-dependent pdes. Preprint at arXiv:2507.03863 (2025).
18. Jin, P., Meng, S. & Lu, L. Mionet: Learning multiple-input operators via tensor product. *SIAM J. Sci. Comput.* **44**, A3490–A3514 (2022).
19. Brandstetter, J., Worrall, D. & Welling, M. Message passing neural PDE solvers. In *International conference on learning representations* (2022).
20. Ronneberger, O., Fischer, P. & Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 234–241 (Springer, 2015).
21. Diab, W. & Al Kobaisi, M. Temporal extrapolation and reliable generalization via 2u-nets deep operator network (2u-deeponet) for time-dependent pdes. In *ECMOR 2024*, vol. 2024, 1–16 (European Association of Geoscientists & Engineers, 2024).
22. Fu, S. et al. Deep learning-based surrogate modeling for underground hydrogen storage. *Adv. Water Resour.* **203**, 105014 (2025).
23. Sofos, F., Drikakis, D., Kokkinakis, I. W. & Spottswood, S. M. Spatiotemporal super-resolution forecasting of high-speed turbulent flows. *Phys. Fluids* **37**, 016124 (2025).
24. Bonneville, C. et al. Accelerating phase field simulations through a hybrid adaptive Fourier neural operator with u-net backbone. *npj Comput. Mater.* **11**, 14 (2025).
25. Michałowska, K., Goswami, S., Karniadakis, G. E. & Riemer-Sørensen, S. Neural operator learning for long-time integration in dynamical systems with recurrent neural networks. In *2024 International joint conference on neural networks (IJCNN)*, 1–8 (IEEE, 2024).
26. Ruiz, R. B., Marwah, T., Gu, A. & Risteski, A. On the benefits of memory for modeling time-dependent pdes. Preprint at arXiv:2409.02313 (2024).

27. He, J. et al. Sequential deep operator networks (s-deeponet) for predicting full-field solutions under time-dependent loads. *Eng. Appl. Artif. Intell.* **127**, 107258 (2024).
28. Hu, Z., Cao, Q., Kawaguchi, K. & Karniadakis, G. E. Deepomamba: State-space model for spatio-temporal pde neural operator learning. *Available at SSRN 5149007* (2025).
29. Hu, Z., Daryakenari, N. A., Shen, Q., Kawaguchi, K. & Karniadakis, G. E. State-space models are accurate and efficient neural operators for dynamical systems. Preprint at arXiv:2409.03231 (2024).
30. Tran, A., Mathews, A., Xie, L. & Ong, C. S. Factorized Fourier neural operators. Preprint at arXiv:2111.13802 (2021).
31. Cornes, R. C., van der Schrier, G., van den Besselaar, E. J. & Jones, P. D. An ensemble version of the e-obs temperature and precipitation data sets. *J. Geophys. Res. Atmos.* **123**, 9391–9409 (2018).
32. Zhu, M., Feng, S., Lin, Y. & Lu, L. Fourier-deeponet: Fourier-enhanced deep operator networks for full waveform inversion with improved accuracy, generalizability, and robustness. *Comput. Methods Appl. Mech. Eng.* **416**, 116300 (2023).
33. Grotch, S. L. & MacCracken, M. C. The use of general circulation models to predict regional climatic change. *J. Clim.* **4**, 286–303 (1991).
34. Bodnar, C. *et al.* Aurora: A foundation model of the atmosphere. Preprint at arXiv:2405.13063 (2024).
35. Pathak, J. et al. Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators. Preprint at arXiv:2202.11214 (2022).
36. Chen, S., Long, G., Jiang, J., Liu, D. & Zhang, C. Foundation models for weather and climate data understanding: A comprehensive survey. Preprint at arXiv:2312.03014 (2023).
37. Kalnay, E. *et al.* The ncep/ncar 40-year reanalysis project. In *Renewable energy*, Vol1_146–Vol1_194 (Routledge, 2018).
38. Ovadia, O., Turkel, E., Kahana, A. & Karniadakis, G. E. Ditto: Diffusion-inspired temporal transformer operator. Preprint at arXiv:2307.09072 (2023).
39. IEA. Net zero roadmap: A global pathway to keep the 1.5 $^\circ$c goal in reach - 2023 update. In *Tech. Rep.*, (International Energy Agency, 2023).
40. Wang, Y. et al. An integrated multi-scale model for CO_2 transport and storage in shale reservoirs. *Appl. Energy* **331**, 120444. https://doi.org/10.1016/j.apenergy.2022.120444 (2023).
41. Pruess, K. & Garcia, J. Multiphase flow dynamics during co_2 disposal into saline aquifers. *Environ. Geol.* **42**, 282–295 (2002).
42. Mahjour, S. K. & Faroughi, S. A. Selecting representative geological realizations to model subsurface co_2 storage under uncertainty. *Int. J. Greenhouse Gas Control* **127**, 103920. https://doi.org/10.1016/j.ijggc.2023.103920 (2023).

Acknowledgements
The authors wish to acknowledge Khalifa University's high-performance computing facilities for providing computational resources. We acknowledge the E-OBS dataset from the EU-FP6 project UERRA (https://www.uerra.eu) and the Copernicus Climate Change Service, and the data providers in the ECA&D project (https://www.ecad.eu).

Author contributions
M.A.K. and W.D. conceptualized and developed the neural network. W.D. wrote the software and generated the visualization. M.A.K. provided supervision. Both authors carried out the formal analysis and validation, and both have written and reviewed the manuscript.

Declarations

Competing interests
The authors declare no competing interests.

Additional information
Supplementary Information The online version contains supplementary material available at https://doi.org/10.1038/s41598-025-16922-5.

Correspondence and requests for materials should be addressed to M.A.K.