Petroleum Science 22 (2025) 1736–1756

Contents lists available at ScienceDirect

Petroleum Science

journal homepage: www.keaipublishing.com/en/journals/petroleum-science

Original Paper

Efficient deep-learning-based surrogate model for reservoir production optimization using transfer learning and multi-fidelity data

Jia-Wei Cui [a, b], Wen-Yue Sun [a, b, *], Hoonyoung Jeong [c, d], Jun-Rong Liu [a, b], Wen-Xin Zhou [a, b]

[a] Key Laboratory of Unconventional Oil & Gas Development, China University of Petroleum (East China), Qingdao, 266580, Shandong, China
[b] School of Petroleum Engineering, China University of Petroleum (East China), Qingdao, 266580, Shandong, China
[c] Department of Energy Systems Engineering, Seoul National University, Seoul 08826, Republic of Korea
[d] Research Institute of Energy and Resources, Seoul National University, Seoul, 08826, Republic of Korea

ARTICLE INFO

Article history:
Received 25 July 2024
Received in revised form
22 February 2025
Accepted 23 February 2025
Available online 25 February 2025

Edited by Yan-Hua Sun

Keywords:
Subsurface flow simulation
Surrogate model
Transfer learning
Multi-fidelity training data
Production optimization

ABSTRACT

In the realm of subsurface flow simulations, deep-learning-based surrogate models have emerged as a promising alternative to traditional simulation methods, especially in addressing complex optimization problems. However, a significant challenge lies in the necessity of numerous high-fidelity training simulations to construct these deep-learning models, which limits their application to field-scale problems. To overcome this limitation, we introduce a training procedure that leverages transfer learning with multi-fidelity training data to construct surrogate models efficiently. The procedure begins with the pre-training of the surrogate model using a relatively larger amount of data that can be efficiently generated from upscaled coarse-scale models. Subsequently, the model parameters are fine-tuned with a much smaller set of high-fidelity simulation data. For the cases considered in this study, this method leads to about a 75% reduction in total computational cost, in comparison with the traditional training approach, without any sacrifice of prediction accuracy. In addition, a dedicated well-control embedding model is introduced to the traditional U-Net architecture to improve the surrogate model's prediction accuracy, which is shown to be particularly effective when dealing with large-scale reservoir models under time-varying well control parameters. Comprehensive results and analyses are presented for the prediction of well rates, pressure and saturation states of a 3D synthetic reservoir system. Finally, the proposed procedure is applied to a field-scale production optimization problem. The trained surrogate model is shown to provide excellent generalization capabilities during the optimization process, in which the final optimized net-present-value is much higher than those from the training data ranges.

1. Introduction

Accurate prediction of subsurface fluid flow is crucial for managing petroleum, natural gas, and groundwater resources. Numerical simulation techniques are often needed to solve subsurface flow equations due to the complexities of geological models and multiphase flow physics. However, the computational resources needed for these simulations can be prohibitive for practical applications, particularly when a large number of full-order simulations are needed for certain optimization and data assimilation tasks. Data-driven surrogate modeling techniques offer a promising approach to significantly reducing the time required for repetitive forward simulations. However, current surrogate modeling methods often require a large number of flow simulations to provide the training data, which can be computationally expensive and limit their use in practice.

In this work, we introduce a novel methodology for constructing deep-learning-based surrogate models that leverage transfer learning with multi-fidelity training data. The surrogate models are

* Corresponding author.
E-mail address: wenyuesunny@outlook.com (W.-Y. Sun).

https://doi.org/10.1016/j.petsci.2025.02.014

trained based on the widely applied U-Net deep learning network with several key improvements (Ronneberger et al., 2015). Firstly, time-varying source/sink terms (i.e.: well controls in this study) are embedded directly into the U-Net network to better capture the corresponding effects on reservoir flow dynamics. Secondly, a model upscaling technique is applied to generate multi-fidelity reservoir models and associated training data. A fine-tuning procedure, the typical method of transfer learning, is then proposed to effectively adjust the model parameters for high-fidelity, or high-resolution, predictions. For the cases considered in this study, 80% of the training runs were generated using efficient coarse-scale models, which led to a roughly 75% reduction in the total computational cost for constructing the deep-learning model. While achieving a significant reduction in computational costs, the final trained model can still provide very accurate dynamic predictions for field pressure and saturation distributions, as well as well rates at different time steps.

The construction of accurate deep-learning-based surrogate models for subsurface flow simulations represents an active area of research. Once trained with sufficient training data, the surrogate models are usually capable of providing very accurate flow predictions, while achieving a speed-up factor of few hundreds to few thousands compared with traditional simulators for subsequent flow simulations (Zhang K. et al., 2021, 2022; Sun, 2020; Kim et al., 2021; Zhong et al., 2021; Nwachukwu et al., 2018). Zhu and Zabaras (2018) introduced the use of a deep convolutional encoder-decoder network to establish relationships between permeability fields and key reservoir characteristics, such as fluid velocity and pressure distribution at the end of the simulation. Mo et al. (2020) extended the encoder-decoder model with input parameters incorporating time-step information for predicting pressure and saturation distributions at different time steps. Tang et al. (2020, 2021) proposed a R-U-Net architecture, which combines the U-Net model with Long Short-Term Memory (LSTM) networks. Zhu and Zabaras (2018), Mo et al. (2020), and Tang et al. (2020) primarily focused on the use of surrogate models for uncertainty quantification, where the well controls were fixed for all different model realizations.

Production optimization in reservoir engineering involves improving the efficiency and profitability of oil and gas extraction operations by maximizing production rates while minimizing operational costs (Kumar et al., 2017; Shang et al., 2019; Zhang Y. et al., 2021). This typically requires the integration of various physical, geological, and economic factors to make informed decisions about reservoir management. However, traditional production optimization processes often require a large number of numerical simulations, bringing a significant computational challenge in oilfield development. Deep learning-based surrogate models can effectively accelerate this process (Jin et al., 2019; Wang et al., 2024). For the production optimization problem, the surrogate model must be able to effectively predict the reservoir state under time-varying well control conditions.

Jin et al. (2019) introduced an approach in which both permeability and well control serve as input features in the model. Zhang K. et al. (2021), Zhong et al. (2021), and Xu et al. (2023) developed similar methods for incorporating well control information. In the abovementioned work, dynamic well-control inputs are broadcasted to a feature matrix in which well-control parameters are non-negative at the matrix coordinates corresponding to well grid blocks. Hence, the well-control feature matrix is very sparse and presents challenges for feature extraction with a limited number of convolutional layers. To better handle well-control input, Kim and Durlofsky (2021, 2023) presented the use of convolutional operations to extract features from geological and well-control input, which were then fed into a Recurrent Neural Network (RNN) for dynamic flow prediction. Jin et al. (2020) proposed an embed-to-control (E2C) framework in which a linear transition model in the low-dimensional hidden space was utilized to handle well-control input. Huang et al. (2023) improved the proposed E2C model by adding a linear transition model to the output layer, allowing for direct prediction of well rates. All of the abovementioned work required a large number, from hundreds to thousands, of detailed flow simulations to provide the training data, which can be computationally infeasible for practical applications.

A promising approach to addressing the requirements of many high-fidelity simulations is to utilize simulation data generated from very efficient coarse-scale models. In this study, we investigate the use of multi-fidelity training data, generated from fine-scale and corresponding upscaled coarse models, to construct deep-learning surrogate models (Christie, 1996; Chen et al., 2008). Geneva and Zabaras (2020) and Meng and Karniadakis (2020) developed composite network structures suitable for training with multi-fidelity data. They adopted transfer learning techniques to adjust surrogate model parameters initially trained using low-fidelity data. Building on these advancements, De et al. (2020) and Song and Tartakovsky (2022) constructed surrogate models using multi-fidelity data to solve two-phase flow problems in 2D Gaussian geomodels. Additionally, Jiang and Durlofsky (2023) have developed surrogate models for reservoir uncertainty quantification. In both the pre-training and fine-tuning phases, the input to the model is a high-fidelity permeability field, and the output is a low-fidelity and high-fidelity saturation field, respectively.

In this study, we develop and apply surrogate models for two-phase oil–water subsurface flow simulations under time-varying well control within the context of well control production optimization. To handle the well-control input, we developed an embedding layer that feeds the well control into the latent space of a traditional U-Net architecture, which was shown to provide superior performance compared with approaches based on traditional treatments of broadcasting well-control input to simulation grid blocks (Jin et al., 2019; Zhang K. et al., 2021; Zhong et al., 2021; Xu et al., 2023). Therefore, the new deep-learning network is referred to as E-U-Net. To train E-U-Net with multi-fidelity data, grid-based upscaling methods were used to generate coarse-scale models from corresponding high-fidelity geomodels. Low-fidelity training data was then generated from these coarse-scale models and used to pre-train the E-U-Net model parameters. A smaller number of high-fidelity training data were then used for model fine tuning. The proposed approach can provide an accurate E-U-Net surrogate model with much less computational cost.

This paper is organized as follows: In Section 2, we introduce the governing equations of the oil–water two-phase subsurface flow problem and reservoir production optimization that is considered in this study. In Section 3, we provide a brief discussion of the E-U-Net network structure, highlighting the specific enhancements and modifications introduced in this study. In addition, model training with multi-fidelity data is provided. In Section 4, we apply the surrogate model to a 3D reservoir model, with detailed discussion on the model accuracy, the effectiveness of using multi-fidelity data, and the impact of the introduced embedding module for well-control input. In Section 5, we apply the proposed surrogate model to a field-scale reservoir model for production optimization. In Section 6, we summarize the study and provide suggestions for future work.

2. Governing equations and reservoir production optimization

2.1. Governing equations

In this work, we will consider a two-phase subsurface flow problem. However, we emphasize that the proposed procedure of constructing deep-learning-based surrogate models can still be applicable for more complex scenarios, such as three-phase flow and compositional models. Regarding two-phase flow, the governing equations for material balance are written as

$$\nabla \cdot (\rho_j \boldsymbol{v}_j) + q_j + \frac{\partial}{\partial t}(\phi \rho_j S_j) = 0, \tag{1}$$

where the subscript j denotes the fluid phase (i.e.: $j = \mathrm{o}$ for oil phase and $j = \mathrm{w}$ for water phase); ρ_j is the phase density; $\boldsymbol{v}_j$ denotes the Darcy velocity of phase j; q_j denotes the source/sink term of phase j; ϕ is the porosity; and S_j is the saturation of phase j. The equation for $\boldsymbol{v}_j$ is written as

$$\boldsymbol{v}_j = -\frac{\boldsymbol{k} k_{rj}(S_j)}{\mu_j(p_j)}\left(\nabla p_j - \rho_j g \nabla z\right), \tag{2}$$

where $\boldsymbol{k}$ is the absolute permeability tensor; k_{rj} represents the relative permeability of phase j; p_j is the pressure of phase j; μ_j is the viscosity of phase j; g is the gravitational acceleration constant; z is the depth. In this study, we ignore the capillary pressure between the two fluid phases, as is common in many conventional reservoir-scale simulations, which leads to $p = p_{\mathrm{o}} = p_{\mathrm{w}}$. Note that k_{rj} and μ_j are nonlinear functions of saturation and pressure of phase j, respectively.

In the context of reservoir simulation, the governing equations, Eqs. (1) and (2), are usually discretized using finite volume methods on geomodels, and the resulting nonlinear equations are typically solved with Newton's method. Define the state vector $\boldsymbol{x} = [\boldsymbol{p}, \boldsymbol{S}_{\mathrm{w}}]$, which contains all the state values on discretized grid blocks, and let $\boldsymbol{u}$ denote the source/sink terms introduced by injectors and producers. The material balance equation shown in Eq. (1) can be written as (see detailed derivation in Aziz (1999))

$$\frac{\partial \boldsymbol{A}(\boldsymbol{x})}{\partial t} + \boldsymbol{F}(\boldsymbol{x}) + \boldsymbol{Q}(\boldsymbol{x}, \boldsymbol{u}) = 0, \tag{3}$$

where $\boldsymbol{A}$, $\boldsymbol{F}$, and $\boldsymbol{Q}$ are the accumulation, flux, and source/sink terms, respectively, which are functions of state vector $\boldsymbol{x}$ and $\boldsymbol{u}$. Discretize Eq. (3) in time and space, the fully-implicit formulation's discrete system can be expressed as

$$\boldsymbol{g}\left(\boldsymbol{x}^{n+1}, \boldsymbol{x}^{n}, \boldsymbol{u}^{n+1}\right) = \boldsymbol{A}\left(\boldsymbol{x}^{n+1}, \boldsymbol{x}^{n}\right) + \boldsymbol{F}\left(\boldsymbol{x}^{n+1}\right) + \boldsymbol{Q}\left(\boldsymbol{x}^{n+1}, \boldsymbol{u}^{n+1}\right), \tag{4}$$

where $\boldsymbol{g}$ represents the residual that we aim to reduce to zero; the superscript n denotes the simulation time step. Newton's method is used to solve Eq. (4), the Jacobian matrix $\boldsymbol{J}$ is constructed as

$$\boldsymbol{J} = \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}, \tag{5}$$

then the iteration can be achieved by

$$\boldsymbol{x}^{n+1} = \boldsymbol{x}^{n} - \left(\boldsymbol{J}^{n+1}\right)^{-1}\left[\boldsymbol{A}\left(\boldsymbol{x}^{n+1}, \boldsymbol{x}^{n}\right) + \boldsymbol{F}\left(\boldsymbol{x}^{n+1}\right) + \boldsymbol{Q}\left(\boldsymbol{x}^{n+1}, \boldsymbol{u}^{n+1}\right)\right]. \tag{6}$$

For detailed explanations and derivations of the above equations, please refer to Aziz (1999) and He et al. (2011).

Let N_{c} denote the number of grid blocks, N_{v} denote the number of primary variables at every grid block, and $N_{\mathrm{T}} = N_{\mathrm{v}} \times N_{\mathrm{v}}$ denote the number of total variables. Then we have $\boldsymbol{J} \in \mathbb{R}^{N_{\mathrm{T}} \times N_{\mathrm{T}}}$. For practical problems, the dimension of $\boldsymbol{J}$ is usually on the order of millions, which leads to significant computational resources and time for each simulation run.

Eqs. (3)–(6) characterize the process of solving the partial differential equation for subsurface flow using a full-order simulator, which can be expressed as calculating the state at time step $n+1$ from the state at time step n, written as

$$\boldsymbol{x}^{n+1} = f\left(\boldsymbol{m}, \boldsymbol{x}^{n}, \boldsymbol{u}^{n+1}\right), \tag{7}$$

where $f(\cdot)$ denotes the numerical simulation from the full-order simulator (i.e.: ECLIPSE used in this study) and $\boldsymbol{m}$ denotes the static parameters such as the permeability field. As mentioned, solving Eq. (7) is computationally expensive and time consuming. In this study, we aim to construct a surrogate model for Eq. (7), which can be written as

$$\widehat{\boldsymbol{x}}^{n+1} = \widehat{f}\left(\boldsymbol{m}, \widehat{\boldsymbol{x}}^{n}, \boldsymbol{u}^{n+1}\right), \tag{8}$$

where $\widehat{f}(\cdot)$ denotes the surrogate model; $\widehat{\boldsymbol{x}}^{n}$ and $\widehat{\boldsymbol{x}}^{n+1}$ are the predicted state vectors from the surrogate model. Note that the same initial conditions are used for the full-order simulator and the surrogate model, therefore we have $\widehat{\boldsymbol{x}}^{0} = \boldsymbol{x}^{0}$. In Section 3, we will describe the method used to construct this surrogate model as well as the use of multi-fidelity simulation data for model training.

2.2. Production optimization using particle swarm optimization

Particle Swarm Optimization (PSO) is a heuristic optimization technique inspired by the social behavior of birds flocking or fish schooling. It is widely employed to solve optimization problems where the objective function is nonlinear, multidimensional, and lacks an explicit mathematical formulation (Poli et al., 2007; Du et al., 2023). In the context of production optimization, PSO can be utilized to optimize well control parameters, such as well rates and bottom-hole pressure, over time to maximize recovery or Net Present Value (NPV). In this work, we have selected NPV as the objective function, and it is calculated as follows:

$$\mathrm{NPV} = \sum_{n=1}^{N_{\mathrm{t}}}\left\{\frac{\Delta t_n}{(1+b)^{\frac{t_n}{365}}}\left(\sum_{i=1}^{N_{\mathrm{o}}} r_{\mathrm{o}} q_{\mathrm{o}}^{n} - \sum_{i=1}^{N_{\mathrm{o}}} c_{\mathrm{pw}} q_{\mathrm{pw}}^{n} - \sum_{i=1}^{N_{\mathrm{w}}} c_{\mathrm{iw}} q_{\mathrm{iw}}^{n}\right)\right\} - \sum_{i=1}^{N_{\mathrm{o}}} c_{\mathrm{o}} - \sum_{i=1}^{N_{\mathrm{w}}} c_{\mathrm{w}}, \tag{9}$$

where N_{t} is the total number of time steps; Δt_n is the time interval for each time step; b is the discount rate; t_n is the cumulative production time; N_{o} and N_{w} are the number of producers and injectors; r_{o} is the price of crude oil; c_{pw} is the cost of handling produced water; c_{iw} is the cost of water injection; q_{o}^{n}, q_{pw}^{n}, and q_{iw}^{n} are the rate of oil production, water production, and water injection at time step n, respectively; c_{o} and c_{w} are the drilling and

completion costs for producers and injectors, respectively.

The rationale for choosing PSO for production optimization lies in its ability to effectively search large, complex, and multidimensional solution spaces. Traditional optimization methods, such as gradient-based techniques, often struggle with oil reservoir models due to their nonlinearity, high dimensionality, and the absence of smooth, continuous gradients (Afshar, 2013; An et al., 2022). PSO, as a population-based global optimization algorithm, does not require gradient information and can efficiently handle the uncertainty and nonlinearity inherent in reservoir modeling and optimization tasks. Note that the choice of optimization algorithm is not the key contribution of this study; hence, only a brief description of PSO will be provided here. For a detailed description of PSO and other more advanced optimization algorithms, please refer to Kochenderfer (2019).

In heuristic optimization techniques, the core lies in the update logic for the next generation. For PSO, the velocity of each particle in the next generation is updated as follows:

$$v_i^{k+1} = w \cdot v_i^k + c_1 \cdot r_1 \cdot \left(p_i^{\text{best}} - x_i^k\right) + c_2 \cdot r_2 \cdot \left(g^{\text{best}} - x_i^k\right), \quad (10)$$

where v_i^{k+1} is the updated velocity of particle i at iteration $k+1$; v_i^k is the current velocity; w is inertia weight; c_1 and c_2 are coefficients of cognitive and social, respectively; r_1 and r_2 are random numbers between 0 and 1; p_i^{best} is the personal best position of particle i; g^{best} is the global best position across all particles; x_i^k is the current position of particle i at iteration k. In this work, we set $w = 0.7$, $c_1 = c_2 = 2.05$. The position updated as follows:

$$x_i^{k+1} = x_i^k + v_i^{k+1}, \quad (11)$$

where x_i^{k+1} is the updated position of particle i at iteration $k+1$. Details of the production optimization procedure use PSO are shown in Table 1.

3. Model architecture and training methods

In this section, we will first describe the E-U-Net architecture and the specific improvements we introduced, then discuss the inputs and outputs of the E-U-Net model. Subsequently, we will describe the use of multi-fidelity data with transfer learning to expedite E-U-Net model training. Finally, we will introduce the loss function used in the model training progress.

Table 1
Pseudo-code for production optimization using PSO.

Algorithm 1: Production optimization procedure use PSO

```
NPV^best = 0
    for i = 1 to N_i
      Initialize v_i^0, x_i^0
      Set p_i^best = x_i^0 and calculate NPV(p_i^best) as Eq. (9)
      if NPV(p_i^best) > NPV^best
        g^best = p_i^best
        NPV^best = NPV(p_i^best)
    end for
    for k = 1 to N_k
      for i = 1 to N_i
        Update the v_i^k and x_i^k as Eqs. (10) and (11)
        Calculate NPV(x_i^k) as Eq. (9) and evaluate particle i
        if NPV(x_i^k) >NPV(p_i^best)
          p_i^best = x_i
        if NPV(p_i^best) > NPV^best
          g^best = p_i^best
          NPV^best = NPV(p_i^best)
      end for
    end for
    return NPV^best, g^best
```

3.1. E-U-net architecture

In this research, we employ the U-Net architecture, as originally developed by Ronneberger et al. (2015). This architecture represents a modified encoder-decoder framework fundamentally based on convolutional neural networks. The adaptability of U-Net has been demonstrated in its recent applications for constructing surrogate models for subsurface flow problems (Zhu and Zabaras, 2018; Sun, 2020). The U-Net architecture comprises encoding and decoding modules, which can effectively capture the spatial features of input information and provide predictions for matrix-like output targets.

The schematic diagram of the U-Net architecture is shown in Fig. 1. The encoding net involves a sequence of convolutional and pooling layers. Convolutional layers are used to extract spatial hierarchies of features from the input features, while pooling layers reduce the spatial dimensions, widen the field of view, and compress the feature representation. The decoding net consists of upsampling layers that increase the resolution of the processed features. This is typically achieved through transposed convolutions or up-scaling operations. Each upsampling step is followed by a convolutional layer to refine the feature maps, enabling the network to reconstruct more detailed spatial information from the compressed feature encoding. The concatenation operation is a critical component of the U-Net architecture. It involves merging the feature maps from the corresponding encoding pathway with the upsampling feature maps in the decoding pathway. This operation provides the decoder with additional context, helping to reconstruct features more precisely in the output map. For more detailed discussions of U-Net, please refer to Ronneberger et al. (2015).

In this study, we introduce an enhanced version of this network, termed Embedding U-Net (E-U-Net), which includes an embedding module to handle time-series well-control input. The detailed architecture of E-U-Net is illustrated in Fig. 2. In the E-U-Net architecture, the input features go through the encoding module to a latent space. Denote the latent-space input matrix as $\boldsymbol{E} \in \mathbb{R}^{x_e \times y_e \times z_e \times n_e}$, where $x_e \times y_e \times z_e$ represents feature dimensions and n_e denotes the number of channels. For a reservoir simulation case that contains n_w wells, the well-control input (similar to boundary conditions) at a specific time step can be assembled into a vector, expressed as $\boldsymbol{u} \in \mathbb{R}^{1 \times n_w}$. The well control is input to the model in two ways: broadcasted input and embedding module. A dedicated embedding module, represented by the gray dashed box in Fig. 2, is introduced to better handle the incorporation of well-control input features. Firstly, a repeat layer is used to expand well-control input into a feature map of size $x_e \times y_e \times z_e \times n_w$. Subsequently, a convolutional layer is applied, with the number of channels changed from n_w to n_u. The corresponding output feature map can then be denoted as $\hat{\boldsymbol{u}}^{n+1} \in \mathbb{R}^{x_e \times y_e \times z_e \times n_u}$.

After merging $\boldsymbol{E}$ and $\hat{\boldsymbol{u}}^{n+1}$, the decoding module transforms the hidden feature tensor into the target tensor, producing the final output: the state predictions for the next time step. Furthermore, within the embedding module, additional dense layers are included to calculate the target flow rates at all wells (i.e.: producers and injectors). We use Global Average Pooling (GAP) to convert the high-dimensional features in the latent space into a 1D vector. GAP computes the average value of each feature map across its spatial

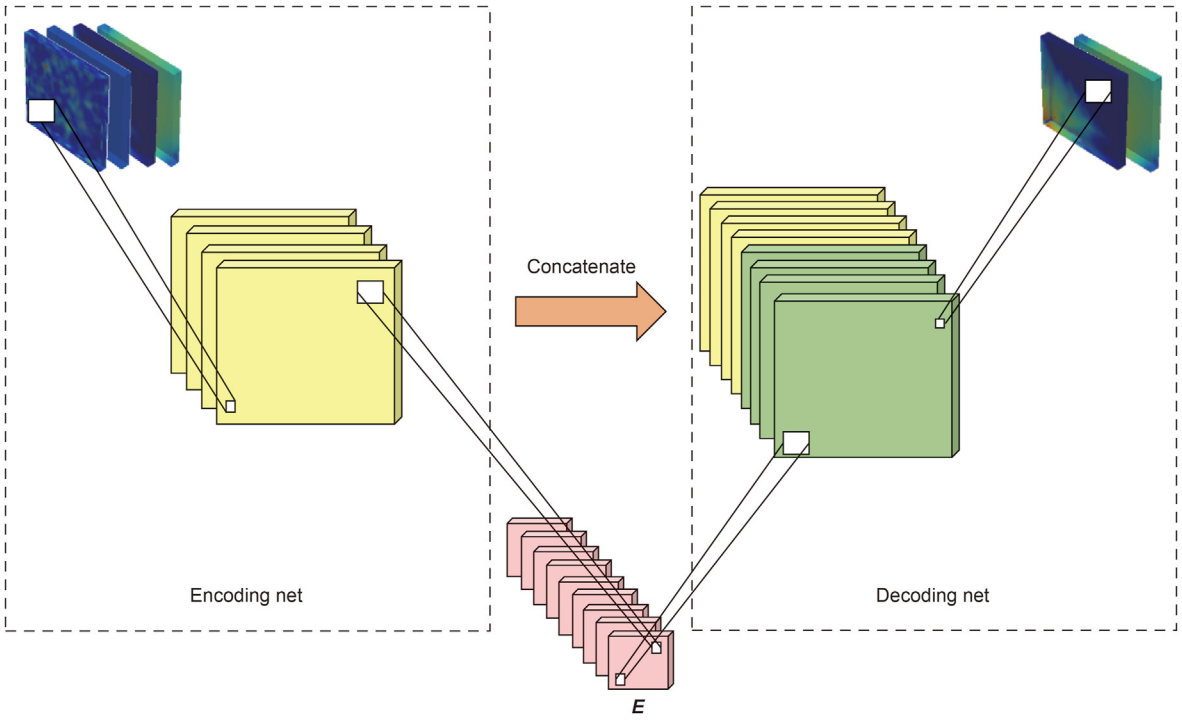


Fig. 1. Schematic diagram of the original 3D U-Net architecture.

dimensions, effectively summarizing spatial information into a single value per feature map. This approach reduces dimensionality while retaining essential global information (Lin, 2013). Then, three layers of fully connected neural networks are used to predict the well rates at the next time step, denoted as $\hat{\boldsymbol{q}}^{n+1}$. A detailed description of the E-U-Net architecture is shown in Table 2. For 3D pooling, all pooling operations utilize a $2 \times 2 \times 1$ kernel to effectively preserve vertical information. The n_x, n_y, and n_z represent the dimension of the input feature and the details of the model input will be described in Section 3.2.

In previous studies, the well-control input was often broadcasted into a high-dimensional feature tensor, with non-zero values

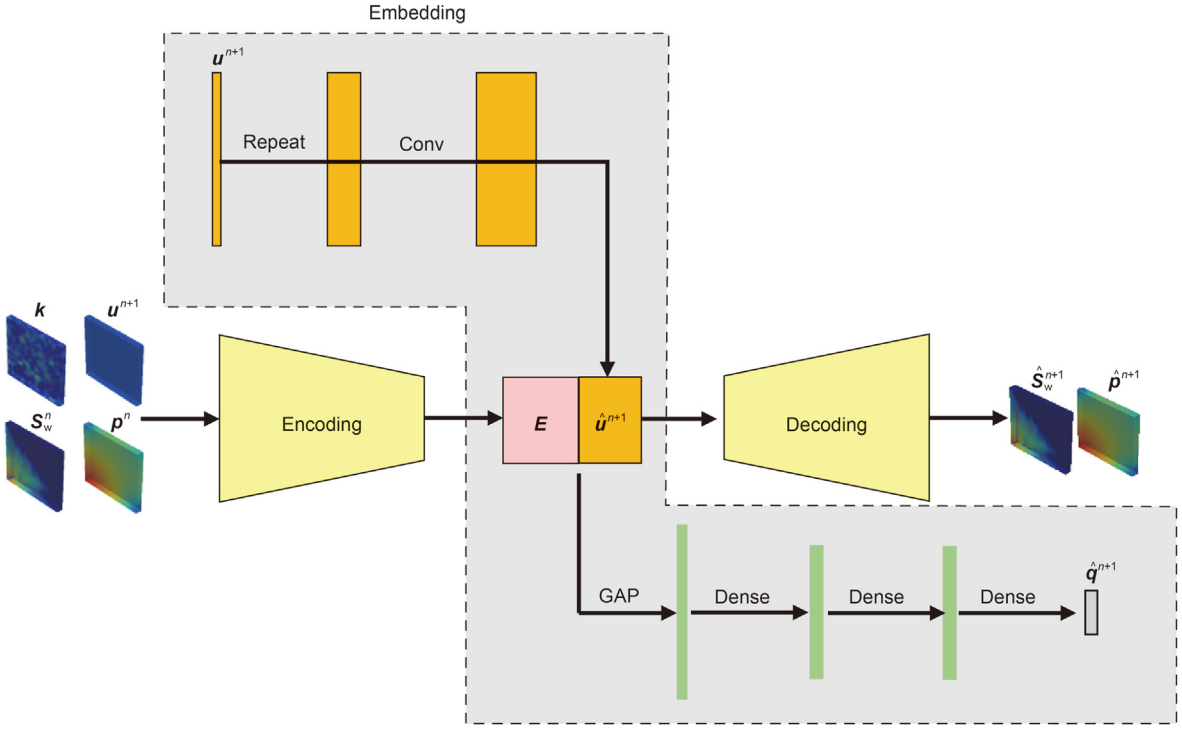


Fig. 2. Schematic diagram of the proposed 3D E-U-Net neural network architecture.

Table 2
Detailed architecture of the proposed 3D E-U-Net.

Layers	Output size
Input 1	$(n_x, n_y, n_z, 4)$
conv, 32 filters of size $3 \times 3 \times 3$, strides 1, pooling	$(n_x/2, n_y/2, n_z, 32)$
conv, 64 filters of size $3 \times 3 \times 3$, strides 1, pooling	$(n_x/4, n_y/4, n_z, 64)$
conv, 128 filters of size $3 \times 3 \times 3$, strides 1, pooling	$(n_x/8, n_y/8, n_z, 128)$
conv, 256 filters of size $3 \times 3 \times 3$, strides 1	$\boldsymbol{E}$ $(n_x/8, n_y/8, n_z, 256)$
Input 2	$(1, n_w)$
Repeat	$(n_x/8, n_y/8, n_z, n_w)$
conv, n_u filters of size $3 \times 3 \times 3$, strides 1	$(n_x/8, n_y/8, n_z, n_u)$
Concatenate $\boldsymbol{E}$	$(n_x/8, n_y/8, n_z, 256 + n_u)$

Layers	Output size	Layers	Output size
conv, 256 filters of size $3 \times 3 \times 3$, strides 1	$(n_x/8, n_y/8, n_z, 256)$	GAP	$256 + n_u$
decov, 128 filters of size $2 \times 2 \times 1$, strides 2	$(n_x/8, n_y/8, n_z, 128)$	Dense	128
decov, 64 filters of size $2 \times 2 \times 1$, strides 2	$(n_x/4, n_y/4, n_z, 64)$	Dense	128
decov, 32 filters of size $2 \times 2 \times 1$, strides 2	$(n_x/2, n_y/2, n_z, 32)$	Output 2	$(1, n_q)$
Output 1	$(n_x, n_y, n_z, 2)$		

at indices corresponding to well locations (Jin et al., 2019; Zhang K. et al., 2021; Zhong et al., 2021; Xu et al., 2023). Such treatment leads to a very sparse well-control input matrix for the constructed surrogate model, especially when the reservoir model is of high-resolution (or high-fidelity). Due to the finite number of CNN layers contained in U-Net, the U-Net model is less affected by the values of the specific grids of the input matrix and cannot effectively propagate the impact of well-control input to the entire simulation field at the next time step, which is critical for accurate pressure and saturation forecasts. Therefore, integrating the provided embedding module after the encoding net represents an important improvement in enabling the global impact of well control on the output. As we will see through the numerical tests presented in Section 4.2, the embedding module helps improve the prediction accuracy of the trained surrogate model, especially when working with high-fidelity cases.

3.2. Model input and output

In this study, the surrogate model takes the static and dynamic reservoir parameters at the current time step as input and then predicts the reservoir's state at the next time step, as represented by Eq. (8). Consider a 3D reservoir model containing $n_x \times n_y \times n_z$ grid blocks and n_w wells (including both producers and injectors), surrogate model inputs and outputs are shown in Table 3. From Table 3, we can see that well-control inputs are fed into the E-U-Net through a broadcasted tensor matrix, as in previous studies (Jin et al., 2019), as well as the discussed embedding module in Section 3.1. Considering the above inputs and outputs, the surrogate model prediction equation during the training phase can be expressed as

$$\widehat{\boldsymbol{x}}^{n+1}, \widehat{\boldsymbol{q}}^{n+1} = \widehat{f}\left(\boldsymbol{k}, \boldsymbol{u}^{n+1}, \boldsymbol{x}^{n}, \theta\right), \tag{12}$$

where θ is the adjustable parameter within surrogate model. Note here that the well rates are also directly predicted as part of the surrogate model output, in comparison with Eq. (8).

Similar to solving partial differential equations with a full-order simulator, the E-U-Net learns mapping between inputs at the current time step and target outputs at the next time step by continuously updating θ during the training process. Fig. 3 illustrates the predictive process implemented by the surrogate model once trained. At the initial time step ($n = 0$), the initial state, $\boldsymbol{x}^0$ and $\boldsymbol{k}$, of the reservoir is known. With a specified well control $\boldsymbol{u}^1$, trained model computes $\widehat{\boldsymbol{q}}^1$ and $\widehat{\boldsymbol{x}}^1$, then $\widehat{\boldsymbol{x}}^1$ acts as the input for the next time step. This iterative process facilitates a sequential characterization of the pressure and saturation distributions within the reservoir, adapting dynamically to variations in well control scenarios over time.

Table 3
Inputs and outputs of the 3D E-U-Net.

Inputs		Dimension
Permeability field $\boldsymbol{k}$		$n_x \times n_y \times n_z \times 1$
Well control $\boldsymbol{u}^{n+1}$	Broadcasted input	$n_x \times n_y \times n_z \times 1$
	Embedding module	$1 \times n_w$
Reservoir state $\boldsymbol{x}^n = [\boldsymbol{p}^n, \boldsymbol{S}_w^n]$		$n_x \times n_y \times n_z \times 2$
Outputs		**Dimension**
Reservoir state $\widehat{\boldsymbol{x}}^{n+1} = [\widehat{\boldsymbol{p}}^{n+1}, \widehat{\boldsymbol{S}}_w^{n+1}]$		$n_x \times n_y \times n_z \times 2$
Well rates $\widehat{\boldsymbol{q}}^{n+1}$		$1 \times n_q$

3.3. Model training with multi-fidelity data

Traditional surrogate models, as demonstrated in Tang et al. (2020) and Kim et al. (2021), demand extensive high-fidelity simulations, often totaling thousands of runs. This is computationally intensive, especially for models with millions of grids. In contrast, our approach leverages transfer learning to address this challenge more efficiently. Transfer learning allows us to transfer knowledge from one task to another, thereby reducing the need for extensive simulations (Pan and Yang, 2009). Specifically, in our case, the source task involves training a surrogate model with low-fidelity data, which is computationally inexpensive but less accurate. The E-U-Net model is initially pre-trained to predict reservoir state variables and well rates for coarse-scale models. The target task, in turn, involves fine-tuning this pre-trained surrogate model with a smaller, more accurate set of high-fidelity data, thereby enhancing the model's predictive accuracy for fine-scale reservoir simulations. By combining both low-fidelity and high-fidelity data, transfer learning enables us to build a model that generalizes well to fine-scale reservoir states, significantly reducing the computational cost of model construction while maintaining high accuracy.

Specifically, the training methodology for E-U-Net using multi-fidelity data is depicted in Fig. 4. The first step involves training E-U-Net parameters using low-fidelity data, with the prediction equation written as

$$\widehat{\boldsymbol{x}}_l^{n+1}, \widehat{\boldsymbol{q}}_l^{n+1} = \widehat{f}\left(\boldsymbol{k}_l, \boldsymbol{u}_l^{n+1}, \boldsymbol{x}_l^{n}, \theta_l\right), \tag{13}$$

where the subscript l stands for low-fidelity; θ_l represents the tunable E-U-Net model parameters. Note that the inputs and outputs in Eq. (13) are all related to coarse-scale reservoir models. During the pre-training process, we employ N_{lf} low-fidelity runs, with each run consisting of N_t time steps. Given that the E-U-Net is designed as a direct mapping of data at previous and next time steps, this results in the creation of $N_{lf} \times N_t$ training samples. These samples are then utilized to train the pre-trained model parameters θ_l. After pre-training, we obtained a pre-trained model that can be used to predict system states for coarse-scale models.

Then, the pre-trained model is fine-tuned using high-fidelity data, with the prediction equation written as

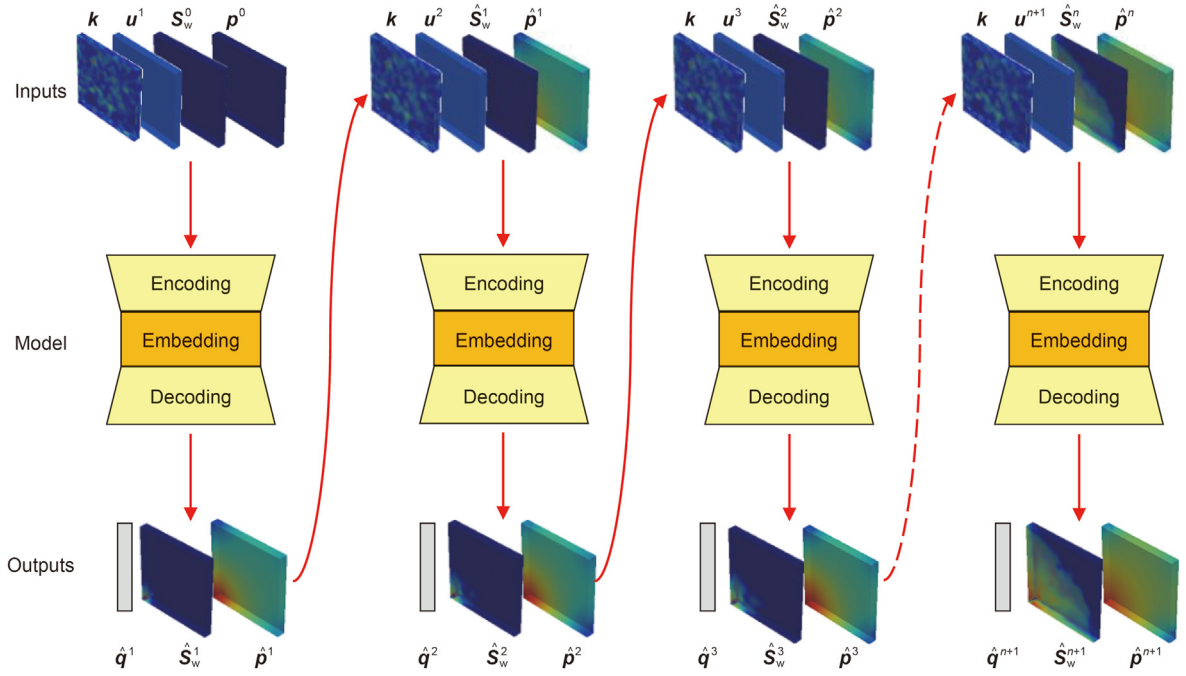


Fig. 3. Schematic diagram of the sequential prediction process of the trained surrogate model. The outputs at the time step n serve as part of the inputs for the time step $n+1$.

$$\hat{\boldsymbol{x}}_{\mathrm{h}}^{n+1}, \hat{\boldsymbol{q}}_{\mathrm{h}}^{n+1} = \hat{f}\left(\boldsymbol{k}_{\mathrm{h}}, \boldsymbol{u}_{\mathrm{h}}^{n+1}, \boldsymbol{x}_{\mathrm{h}}^{n}, \theta_{\mathrm{h}}\right) \tag{14}$$

where the subscript h denotes high-fidelity. During the fine-tuning process, we use N_{ft} high-fidelity models, and similar to the pre-training process, $N_{\mathrm{ft}} \times N_{\mathrm{t}}$ training samples are created. After the fine-tuning process, we can obtain updated model parameters θ_{h} that are suitable for generating predictions for fine-scale models. We emphasize that the proposed fine-tuning strategy is a simple but effective method of transfer learning, as the E-U-Net is built on a convolutional neural network framework, where the surrogate model parameters primarily consist of fixed-size kernel operations that are invariant w.r.t the dimensions of input and output matrices. This means the trainable parameters represented by θ_{h}

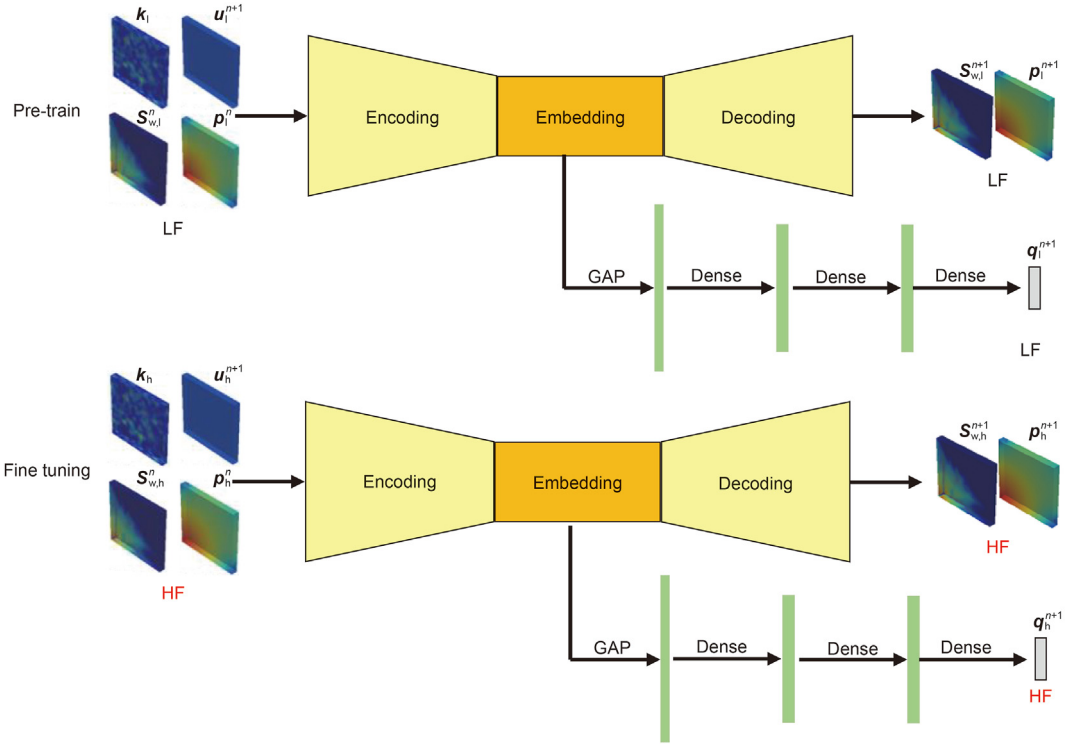


Fig. 4. E-U-Net model training through a two-step process with multi-fidelity training data. Here, LF stands for low-fidelity data and HF stands for high-fidelity data.

have the same dimension as those of θ_l. The trained kernel parameters from low-fidelity training data serve as good initial estimates for training the final high-fidelity E-U-Net surrogate model. In this paper, both the pre-training and fine-tuning processes involved model training for 300 epochs, and the amount of high and low-fidelity data used in pre-training and fine-tuning is described in detail in Section 4.3.

During the training process, the choice of the loss function is very important. In our model, the losses during the training process contain saturation loss (L_S), pressure loss (L_p), and well rate loss (L_q). These losses are denoted as

$$L_S = \frac{1}{N_{run}} \frac{1}{N_t} \sum_{i=1}^{N_{run}} \sum_{n=1}^{N_t} \| \boldsymbol{S}_i^n - \widehat{\boldsymbol{S}}_i^n \|_2^2, \tag{15}$$

$$L_p = \frac{1}{N_{run}} \frac{1}{N_t} \sum_{i=1}^{N_{run}} \sum_{n=1}^{N_t} \| \boldsymbol{p}_i^n - \widehat{\boldsymbol{p}}_i^n \|_2^2, \tag{16}$$

$$L_q = \frac{1}{N_{run}} \frac{1}{N_t} \sum_{i=1}^{N_{run}} \sum_{n=1}^{N_t} \| \boldsymbol{q}_i^n - \widehat{\boldsymbol{q}}_i^n \|_2^2, \tag{17}$$

where N_{run} is the total number of simulation runs to generate the training samples; N_t is the total number of time steps for each simulation. The total loss function used in this study is

$$L_T = \lambda L_S + \beta L_p + \delta L_q, \tag{18}$$

where the hyperparameters λ, β, and δ are specified weights for each loss component. All the training samples are preprocessed using the standard min-max normalization technique, and the three hyperparameters λ, β, and δ were all set to 0.33 for cases considered in this study. The training and evaluation processes were performed on a computational environment featuring a 13th Gen Intel Core i9-13900K CPU and an NVIDIA GeForce RTX 4070 Ti GPU.

To generate multi-fidelity training data, a simple grid-based mean upscaling technique is applied (Christie, 1996; Chen et al., 2008). Essentially, the fine grids are grouped into a number of coarse grid cells, with coarse-scale properties obtained by corresponding average fine-scale properties. We perform upscaling on heterogeneous permeability fields. Fig. 5 illustrates the distribution of permeability ($\ln k$) after upscaling, calculated by

$$k_{coarse}(i,j,k) = \frac{1}{r^2} \sum_{a=0}^{r-1} \sum_{b=0}^{r-1} k_{fine}(ri+a, rj+b, k), \tag{19}$$

where k_{coarse} and k_{fine} are the coarse-scale and fine-scale permeability fields, respectively; r is the upscaling factor; and i, j, and k are the grid coordinates of the coarse-scale model, respectively. Note that more advanced upscaling methods can also be utilized, such as those upscaling algorithms considering single-phase or multi-phase equations (Li et al., 2014; Li and Durlofsky, 2016). The use of more advanced upscaling methods can potentially improve the accuracy of the E-U-Net surrogate model when trained with multi-fidelity data. Also note that the proposed two-step training procedure shown in Fig. 4 can be easily extended to a multi-step training procedure with multiple levels of upscaling for further reduction of computational cost.

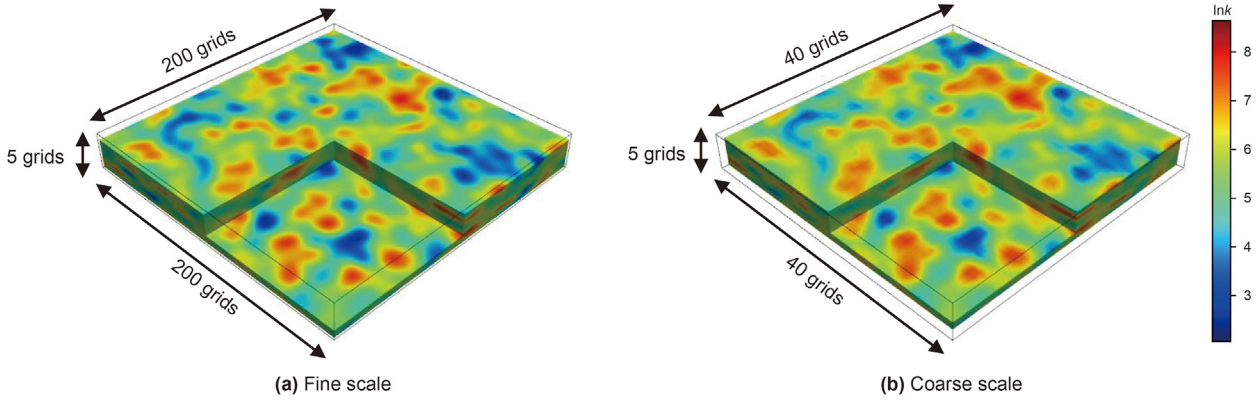
The trained surrogate model can replace the traditional numerical simulation process by efficiently predicting the reservoir state and well rates under different well control scenarios. Throughout the production optimization process, the surrogate model enables fast NPV predictions, shown in Eq. (9), under varying well control conditions. It is important to emphasize that only the trained surrogate model is used to calculate the NPV during production optimization, and the multi-fidelity data is solely employed during the model training procedure. The workflow for constructing the surrogate model using multi-fidelity data and applying it to production optimization is illustrated in Fig. 6.

4. Numerical tests

In this section, we first introduce the problem setup. Subsequently, we discuss the effect of the well-control embedding module on model accuracy. The sensitivity of model fine-tuning using different numbers of high-fidelity simulations is also presented. Finally, we analyze the computational costs associated with our proposed procedure for building surrogate models.

4.1. Problem setup

In this section, we consider a 3D oil–water subsurface flow problem. The fine-scale simulation model is represented on $200 \times 200 \times 5$ Cartesian grid blocks, with each block of size 5 m × 5 m × 3 m. The coarse-scale model is represented on $40 \times 40 \times 5$ grid blocks, with an upscaled factor of five. The permeability field is generated by the sequential Gaussian method (Müller et al., 2020), with a mean of the natural logarithmic values of permeability ($\ln k$)



(a) Fine scale **(b)** Coarse scale

Fig. 5. Comparison of fine-scale and coarse-scale permeability fields before and after simple model upscaling.

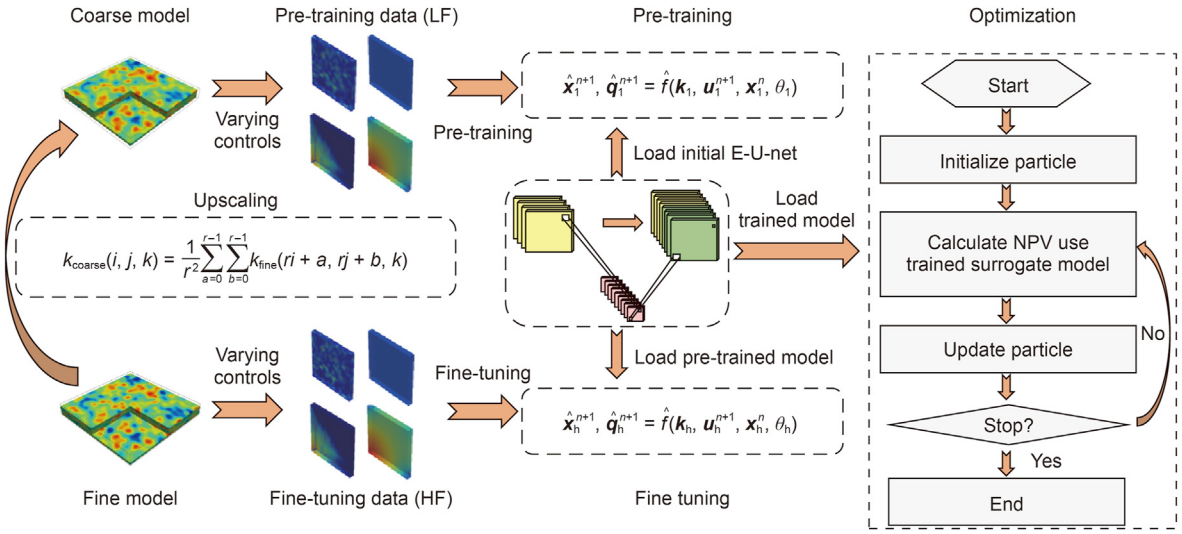


Fig. 6. Workflow for training a surrogate model using multi-fidelity data and applying it to production optimization.

of 5 mD and a standard deviation of ln k to be 1.6. The permeability field and well locations are illustrated in Fig. 7, where the prefixes I and P denote an injector and a producer, respectively.

The oil–water relative permeability curves used in this case are shown in Fig. 8, which are calculated based on standard Corey functions, written as

$$k_{\mathrm{rw}} = (k_{\mathrm{rw}})_{S_{\mathrm{orw}}} \left[\frac{S_{\mathrm{w}} - S_{\mathrm{wc}}}{1 - S_{\mathrm{wc}} - S_{\mathrm{orw}}} \right]^{n_{\mathrm{w}}}, \tag{20}$$

$$k_{\mathrm{ro}} = (k_{\mathrm{ro}})_{S_{\mathrm{wc}}} \left[\frac{1 - S_{\mathrm{w}} - S_{\mathrm{orw}}}{1 - S_{\mathrm{wc}} - S_{\mathrm{orw}}} \right]^{n_{\mathrm{o}}}, \tag{21}$$

where the parameters are as follows: $S_{\mathrm{orw}} = 0.2$, $S_{\mathrm{wc}} = 0.2$, $(k_{\mathrm{rw}})_{S_{\mathrm{orw}}} = 1$, $(k_{\mathrm{ro}})_{S_{\mathrm{wc}}} = 1$, $n_{\mathrm{w}} = 1.5$, $n_{\mathrm{o}} = 3$. Please refer to Sun et al. (2017) for a detailed definition of each parameter in Eqs. (20) and (21).

Initial oil and water saturations are set to 0.8 and 0.2, respectively. Reservoir porosity is set to a constant value of 0.15. The initial reservoir pressure is 400 bar. The range of injection bottom-hole pressure (BHP) is set between 550 and 650 bar, while the production well BHPs are maintained within a range of 250–350 bar. BHP controls in all wells are adjusted every 60 days for a total of 20 time steps. Hence, the total simulation time is 1200 days. Fig. 9 shows the BHP controls for one case, in which the values were randomly sampled for each time step within specified ranges.

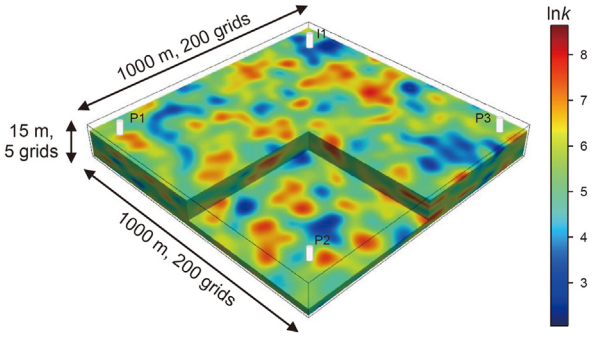


Fig. 7. Distribution of permeability (in natural logarithmic scale) and well locations (I and P stand for injector and producer, respectively).

4.2. Impact of well-control embedding module

In this section, we discuss the impact of adding the embedding module for well-control input by setting different values for n_{u}, which is the channel number for the well-control feature after convolution (shown in Table 2). A total of 300 training runs and 100 testing runs were performed by varying the well controls randomly within the specified ranges that are provided in Section 4.1. For each training or testing run, the pressure/saturation states and well rates were recorded for 20 time steps, with each time step of 60 days.

We will evaluate the impact of the well-control embedding module when constructing surrogate models for the low-fidelity model (of shapes $40 \times 40 \times 5$) and the high-fidelity model (of shapes $200 \times 200 \times 5$), respectively. The goal here is to present the impact of adding dedicated treatments for time-series well-control input when building surrogate models for high-fidelity and low-fidelity model scenarios. For each scenario, we generated two surrogate models: one without the embedding module, U-Net (i.e.: $n_{\mathrm{u}} = 0$) and one with the embedding module, E-U-Net (i.e.: $n_{\mathrm{u}} = 32$). Note that the surrogate models here were trained using single-fidelity data, results using multi-fidelity data will be presented in Section 4.3.

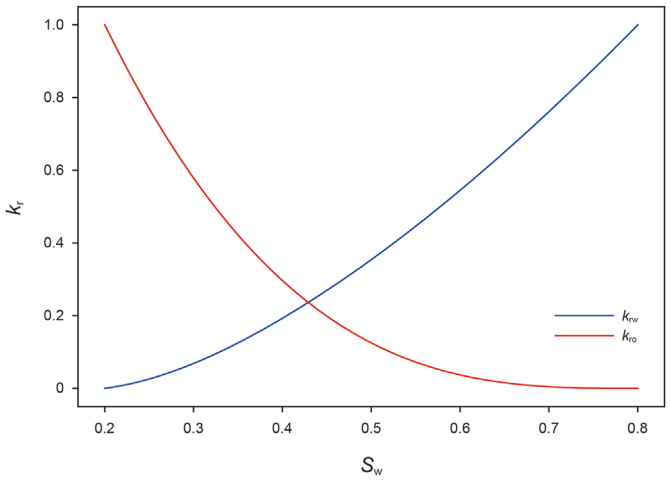


Fig. 8. Oil–water relative permeability curves.

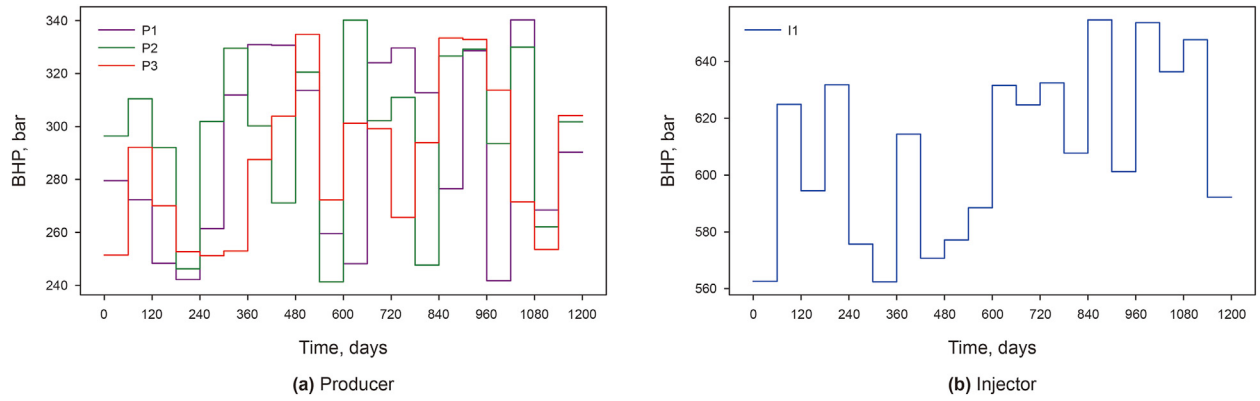


Fig. 9. Well bottom-hole pressure (BHP) controls for injector and producers.

Fig. 10 shows the predicted water saturation and pressure states at the last time step for one randomly selected test run w.r.t the low-fidelity model. Fig. 10(a) shows the comparison of the water saturation results. The different rows stand for results from the surrogate model based on the U-Net and E-U-Net models, respectively. The different columns present the result of the full-order simulation, the surrogate model, and the error of both, respectively. It is clear that improved predictions can be obtained for a trained surrogate model using E-U-Net, as shown in the third column. The maximum saturation prediction errors were reduced from roughly 0.06 to 0.03 with the introduction of the well-control embedding module. Fig. 10(b) shows the results comparison for corresponding pressure predictions. Again, we can clearly see that the surrogate model built using E-U-Net presents improved pressure predictions. The overall prediction errors are very small and span from 0 to 4 bar, which are expected due to the small model size.

Fig. 11 shows the comparison results w.r.t a high-fidelity model

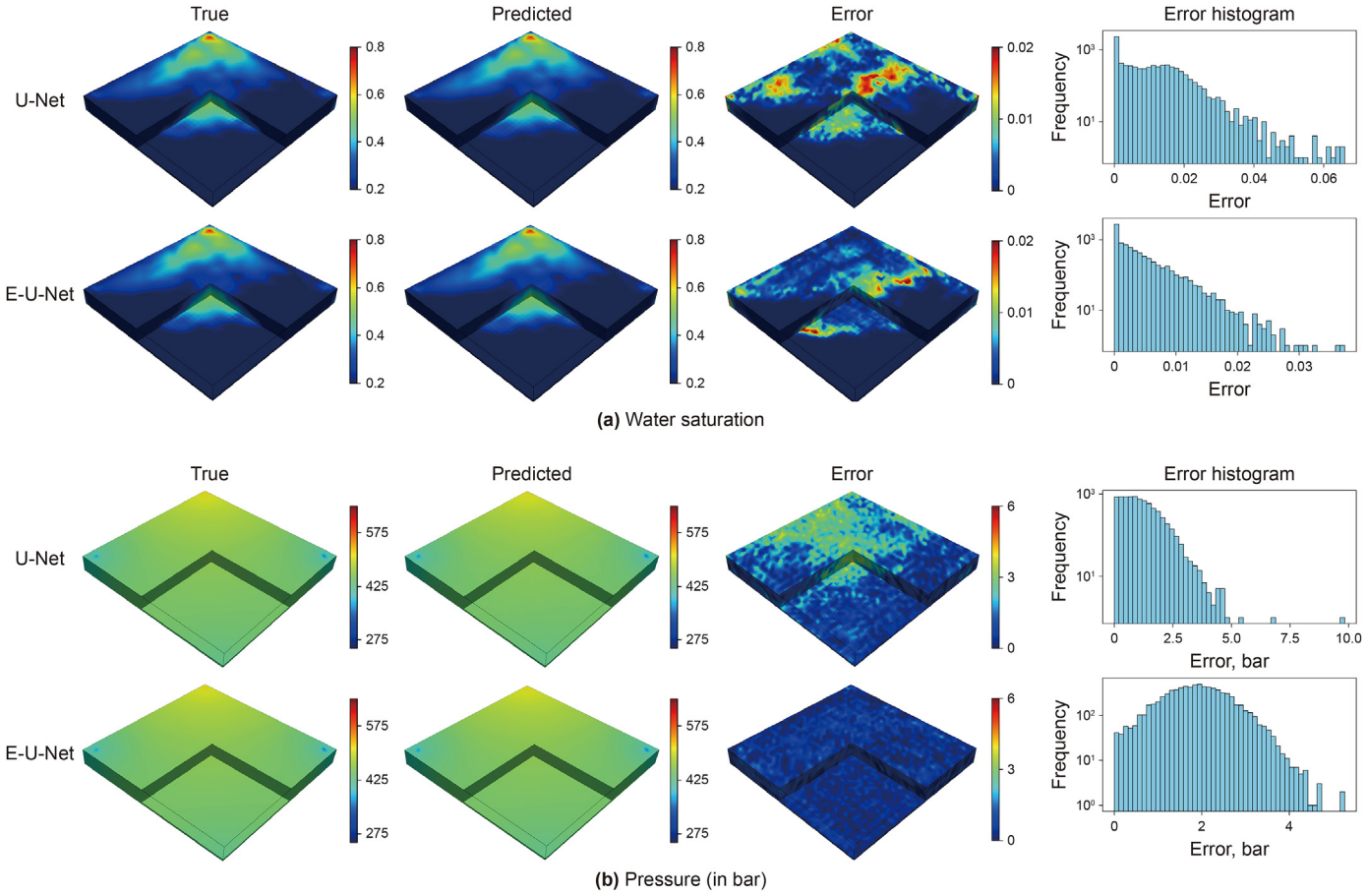


Fig. 10. Water saturation and pressure at the last time step of the simulation for the low-fidelity model. The first two columns present predictions from the full-order simulation and the surrogate model, respectively. The third column shows the absolute prediction error, and the last column is the error histogram. The first row stands for results from the surrogate model using U-Net ($n_u = 0$), and the second row is for results with the proposed E-U-Net ($n_u = 32$).

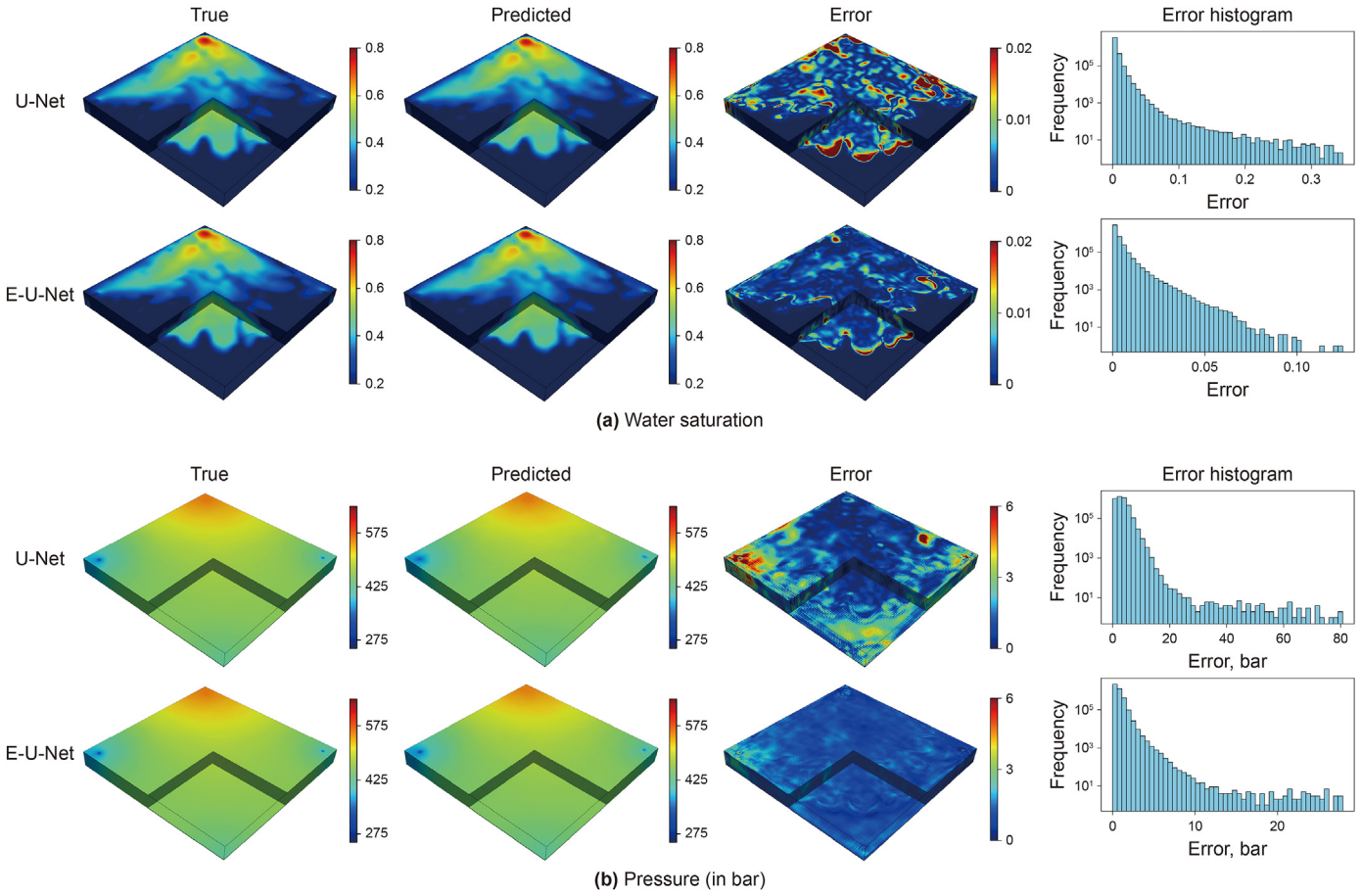


Fig. 11. Water saturation and pressure at the last time step of the simulation for the high-fidelity model. The figure layout is the same as that in Fig. 10.

corresponding to the coarsened model used in Fig. 10. The figure layout is the same as that in Fig. 10. Here we can clearly see that the trained surrogate model using E-U-Net provides superior results in comparison with U-Net (third and fourth columns in Fig. 11(a) and (b)). For example, the maximum error for water saturation prediction reduces from about 0.3 to 0.1, and the maximum error for pressure prediction reduces from about 80 to 25 bar. In addition, comparing the results in Figs. 10 and 11, we can see that the

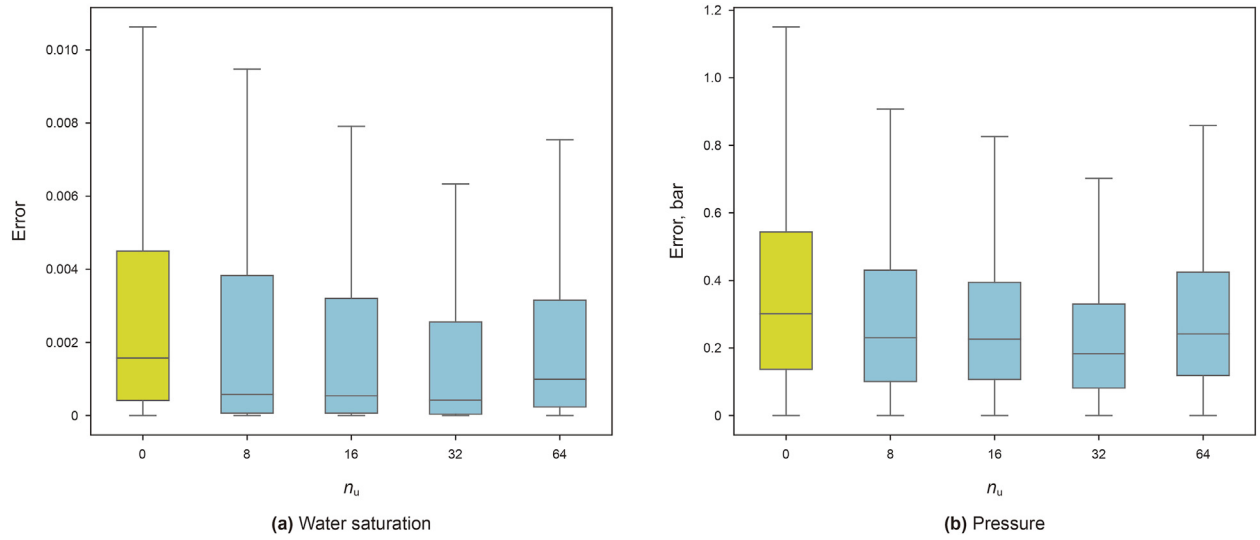


Fig. 12. Box plots of prediction errors for water saturation and pressure fields from the trained surrogate model with different channel numbers of the well-control embedding module (n_u). Note that $n_u = 0$ (filled in yellow) means the surrogate model is trained using U-Net. Each box represents the P10 (bottom line), P25 (bottom of the box), P50 (middle line), P75 (top of the box), and P90 (top line) results. The results here are for the low-fidelity model ($40 \times 40 \times 5$ grid blocks).

prediction errors from a trained surrogate increase when dealing with a higher-fidelity model. Such observations indicate that it tends to be more challenging to build accurate deep-learning-based surrogate models for fine-scale simulation cases.

Another important observation is that, by comparing the results shown in Figs. 10 and 11, the E-U-Net model provides larger accuracy improvements, compared with the results from the U-Net model, when dealing with the high-fidelity model. For example, the ranges of prediction error histograms of pressure are similar between the U-Net and E-U-Net models, as shown in the fourth column of Fig. 10(b) for the low-fidelity case, while the corresponding reduction of the maximum error is from about 80 to 25 bar for the high-fidelity case, as shown in the fourth column of Fig. 11(b). Similar results are observed regarding saturation predictions. These observations further demonstrate the importance of having a specific embedding module to handle time-series well-control input, as mentioned in Section 3.1.

Figs. 10 and 11 show comparison results for one fine-scale case and the corresponding coarse-scale case. Here we will present statistics of the comparison results w.r.t all 100 randomly generated test runs, which are not included in any of the training runs. The absolute errors between the true predictions and surrogate-model predictions are calculated for both pressure and saturation fields, written as

$$\delta_{i,j}^{\mathrm{S},n} = \left| S_{i,j}^{n} - \widehat{S}_{i,j}^{n} \right|, \tag{22}$$

$$\delta_{i,j}^{\mathrm{p},n} = \left| p_{i,j}^{n} - \widehat{p}_{i,j}^{n} \right|, \tag{23}$$

where $\delta_{i,j}^{\mathrm{S},n}$ and $\delta_{i,j}^{\mathrm{p},n}$ are the absolute error between true and surrogate-model results for the test run i, grid block j, and time step n. For the problem considered here, we have $i = 1,2,\ldots,100$, $n = 1, 2,\ldots,20$, and $j = 1,2,\ldots,N_{\mathrm{c}}$. For the fine-scale test case, we have $N_{\mathrm{c}} = 200 \times 200 \times 5 = 200000$. For the corresponding coarse-scale test case, we have $N_{\mathrm{c}} = 40 \times 40 \times 5 = 8000$.

Fig. 12 presents the error statistics, in the form of box plots, for water saturation and pressure predictions for the low-fidelity model from the trained E-U-Net model with different channel numbers of the well-control embedding module (n_{u} in Table 2). Each box represents the P10 (bottom line), P25 (bottom of the box), P50 (middle line), P75 (top of the box), and P90 (top line) of calculated errors in Eq. (22) or Eq. (23). Fig. 13 presents the error statistics regarding the high-fidelity model. Comparing Figs. 12 and 13, it is clear that the prediction errors for both saturation and pressure fields are systematically larger when dealing with high-fidelity models, though the overall errors are quite small for E-U-Net predictions (e.g.: prediction errors are generally below 0.02 for saturation fields and below 3 bar for pressure fields).

From Figs. 12 and 13, we can again see that the E-U-Net model ($n_{\mathrm{u}} = 0$) clearly provides more accurate predictions, compared with the U-Net model results where $n_{\mathrm{u}} = 0$, especially for the high-fidelity model. These results demonstrate the effectiveness of adding a dedicated well-control embedding module. Based on the numerical results presented in Figs. 12 and 13, we chose $n_{\mathrm{u}} = 32$ as the optimal hyperparameter number and will use this setting for the following sections.

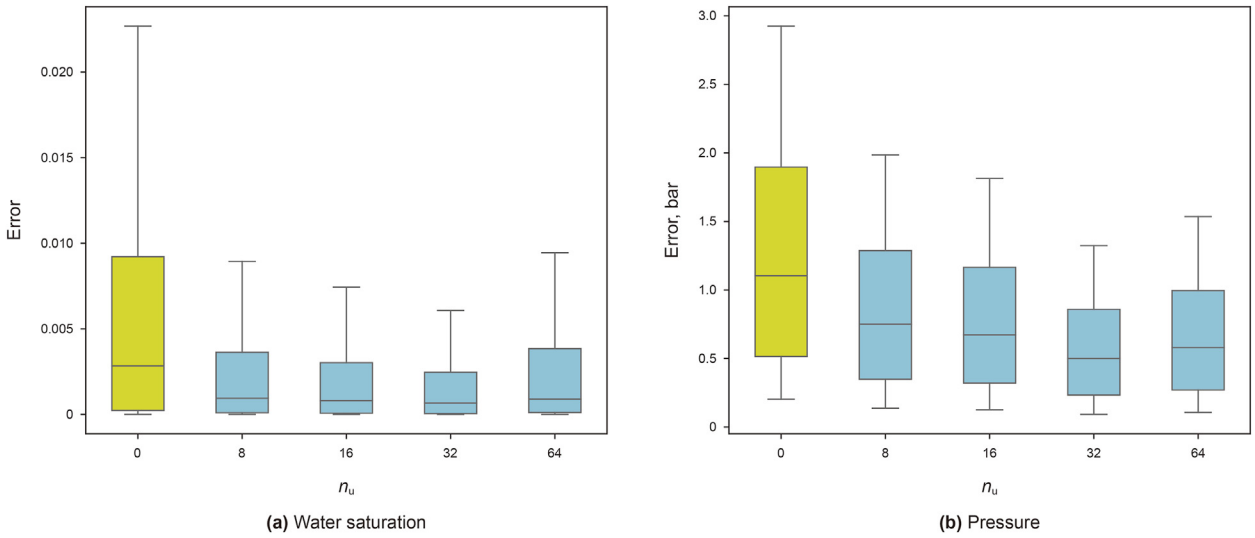
4.3. Transfer learning with multi-fidelity data

In this section, we discuss the prediction accuracy of surrogate models constructed using transfer learning combined with multi-fidelity data. We compare the prediction accuracy of surrogate models constructed using multi-fidelity data with those constructed using only high-fidelity data. Table 4 shows the number of training runs used to construct the surrogate models FT (N_{ft}) and HF (N_{hf}).

From Section 4.2, we can see that when the model is trained with single-fidelity data, a high prediction accuracy can be achieved with 300 training runs. Therefore, the number of low-fidelity runs used in pre-training is set to 300. The subsequent methods of constructing surrogate models using multi-fidelity all use 300 low-fidelity runs to pre-train the model. In the sections below, HF (N_{hf}) represents a surrogate model that is trained with N_{hf} single-high-fidelity runs, and FT (N_{ft}) represents a surrogate model that is trained with N_{ft} high-fidelity runs and 300 low-fidelity runs.

Table 4
Number of training runs used by different models.

Model	Training data	High-fidelity runs	Low-fidelity runs
FT (N_{ft})	Multi-fidelity	N_{ft}	300
HF (N_{hf})	Single-fidelity	N_{hf}	–



(a) Water saturation **(b)** Pressure

Fig. 13. Box plots of prediction errors regarding the high-fidelity model (200 × 200 × 5 grid blocks). The colors and lines here have the same meanings as in Fig. 12.

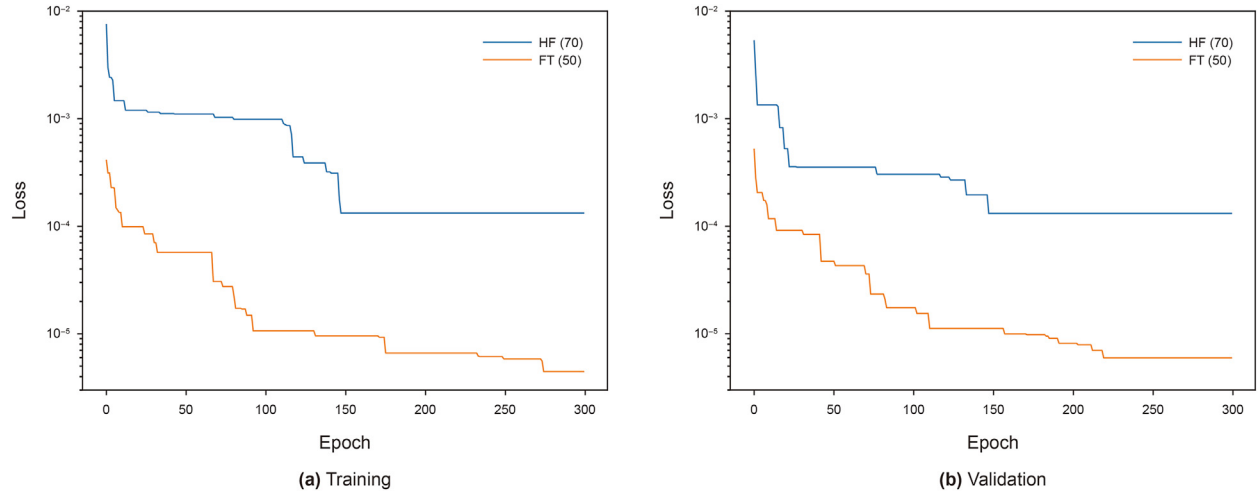


Fig. 14. The evolution of training and validation losses for different models. Here, the FT (50) model used 50 high-fidelity and 300 low-fidelity training runs, while the HF (70) model used only 70 high-fidelity training runs.

Table 5
Comparative analysis of time consumption in surrogate-model construction using single-high-fidelity and multi-fidelity data.

	Time consumption, h		
	HF (70)	HF (300)	FT (50)
Data generation	11.81	50.62	12.48 (4.05 + 8.43)
Training	2.73	10.42	0.93
Fine-tuning	–	–	1.22
Total	14.54	61.04	14.63

Fig. 14 shows the training and validation loss evolutions while training FT (50) and HF (70). Note that the time consumed by 300 low-fidelity model simulations is close to the time consumed by 20 high-fidelity model simulations, so we use HF (70) as the comparison model. The detailed time consumption of surrogate model construction will be shown in Table 5. At each epoch, the current minimum loss values are presented. For model training, the learning rate was set to 0.0001, the batch size was 10, and the Adaptive Moment Estimation (Adam) algorithm was used for network optimization (Kingma, 2014). Here we can clearly see that the results of FT (50) show much faster convergence and smaller convergence loss compared to those of HF (70). The validation loss reaches a final convergence value of approximately 10^{-5} for FT (50), which is almost 10 times smaller than that for HF (70). The comparison results demonstrate the effectiveness of using computationally efficient low-fidelity runs for E-U-Net model pre-training.

Fig. 15 presents the randomly generated well control parameters used in a testing run, while Fig. 16 illustrates the predicted water saturation and pressure states at various selected time steps during this run. Fig. 16(a) shows the comparison results of the water saturation prediction. It is clear that FT (50) presents significantly improved saturation prediction than HF (70). The maximum error for water saturation prediction reduces from about 0.18 to 0.07, representing a reduction of nearly 60%. Fig. 16(b) shows the comparison of the corresponding pressure predictions. Again, we can clearly see that better predictions can be obtained from FT (50) than those from HF (70). The maximum of the error for pressure prediction reduces from about 65 to 25 bar, resulting in a reduction of prediction error by nearly 60%.

Fig. 17 shows the predictive accuracy of the well rate at each time step. It can be observed that FT (50) is more accurate than HF (70) in predicting the well rate at each time step. The rate predicted of FT (50) can be well matched with the results of full-order simulations, whereas HF (70) has a large prediction error in several data points. We emphasize that accurate prediction of the well rates at each time step is critical for the use of surrogate models for field development optimization.

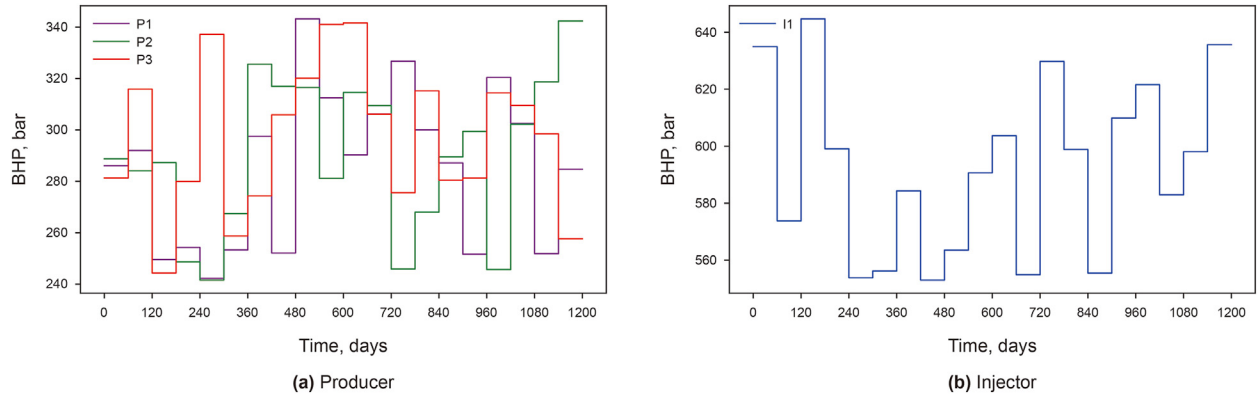


Fig. 15. Well control setting for a testing run.

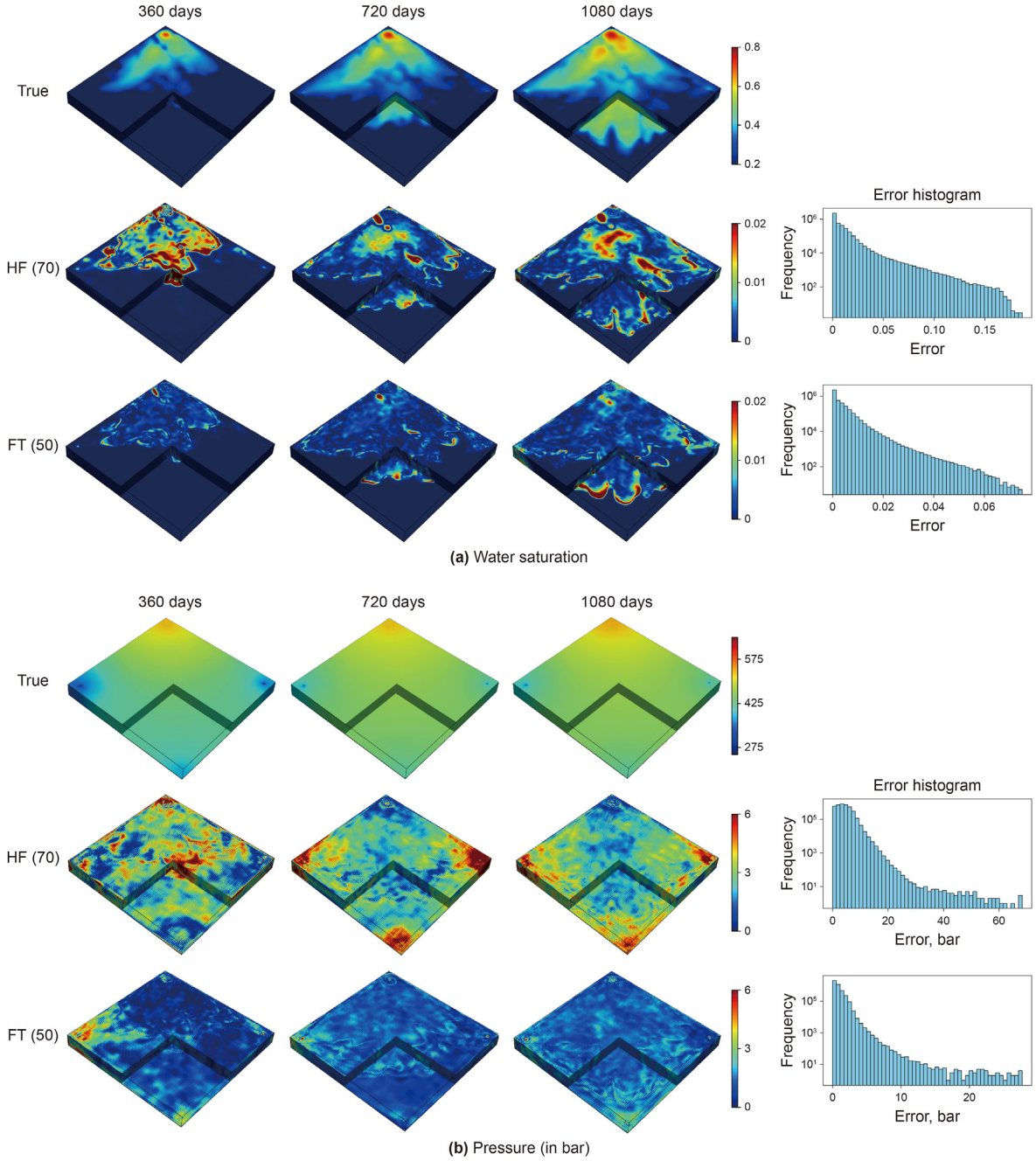


Fig. 16. Prediction of HF (70) and FT (50) surrogate models on water saturation and pressure at selected time steps. For each subplot, the first row shows the results of full-order simulation, the second and third rows show the prediction errors of HF (70) and FT (50), respectively. The first three columns indicate the results at 360, 720, and 1080 days, respectively, and the last column displays the histogram of the distribution of prediction errors for all three time steps shown.

Based on the analysis above, it is apparent that FT (50) is able to accurately predict the pressure, water saturation, and well rate at each time step under time-varying well control conditions with a small number of high-fidelity training runs. To further discuss the predictive capabilities of the fine-tuned model, we will present statistics on the prediction errors after fine-tuning with varying numbers of high-fidelity runs. We will provide statistics on the comparison results w.r.t all 100 randomly generated test cases. The prediction errors for water saturation and pressure are calculated in the same manner as described in Eqs. (22) and (23). The well rate errors are calculated in the same way:

$$\delta_{i,j}^{\mathrm{q},n} = \left| q_{i,j}^{n} - \widehat{q}_{i,j}^{n} \right|, \tag{24}$$

where $\delta_{i,j}^{\mathrm{q},n}$ is the absolute error between true and surrogate-model rates for the test run i, well rate j, and time step n. For the problem considered here, we have $i = 1,2,\ldots,100$, $n = 1,2,\ldots,20$, and $j = 1,2,\ldots,n_{\mathrm{q}}$. For the test case, we have one injector and three producers, so $n_{\mathrm{q}} = 1 \times 1 + 3 \times 2 = 7$.

Figs. 18 and 19 show the error statistics, in the form of box plots, for water saturation, pressure, and well rate for different surrogate models. The lines here have the same meanings as in Fig. 12. HF

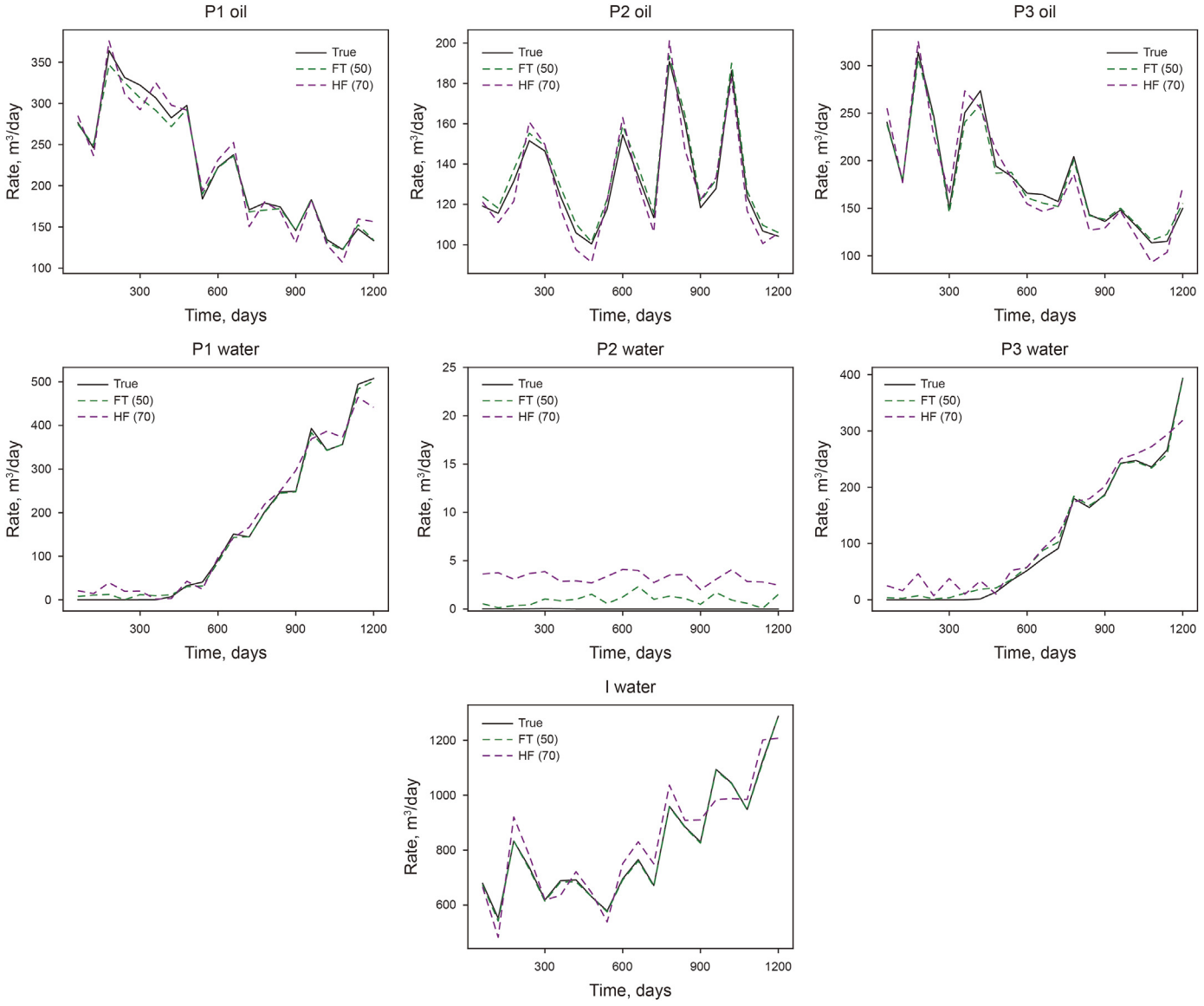


Fig. 17. Well rate predictions of surrogate models FT (50) and HF (70). The black line represents true results obtained from full-order reference simulation, the green and purple dashed lines are results from FT (50) and HF (70), respectively.

(N_{hf}) and FT (N_{ft}) here have the same meanings as in Table 4**.** The models trained with single-fidelity data are filled in yellow and those trained with multi-fidelity data are filled in blue.

From Figs. 18 and 19, we can see that the prediction errors for water saturation, pressure, and well rate gradually decrease as N_{ft} increases, approaching the prediction results of HF (300). In the case of water saturation prediction, as shown in Fig. 18(a), HF (70) has a median error of 0.0042, when $N_{ft} \geq 50$, the median error of FT (N_{ft}) stabilizes at 0.0011 and approaches the prediction results of HF (300) at 0.0009. In terms of pressure prediction, as shown in Fig. 18(b), the prediction error reduces substantially when comparing FT (5) to HF (70), which demonstrates the effectiveness of the proposed two-step training procedure. In addition, for the surrogate model FT (N_{ft}), the pressure prediction errors stabilize after $N_{ft} = 50$ with a median error of 0.73 bar and are close to those from using a large number of high-fidelity runs for training, i.e.: HF (300) with a median error of 0.51 bar.

Similar observations are shown in Fig. 19 for well rate predictions: FT (5) shows clear improvements compared to the results of HF (70), which has a median error of 8.52 m^3/day. With only 50 high-fidelity training runs, the median error of the FT (50) model is 1.92 m^3/day, which is similar to the error level of the HF (300) model at 1.86 m^3/day. Note that each high-fidelity run here is much more expensive than the corresponding low-fidelity run.

We now present statistics on the time required to construct the surrogate models HF (300) and FT (50), which have similar levels of prediction accuracy as shown in Figs. 18 and 19. The time of the HF (70) model, which was used as a comparison model, was also estimated. Table 5 shows the time required for building models using single-high-fidelity data or multi-fidelity data. The computational time for the entire model construction process primarily consists of two parts: data generation and model training. Each high-fidelity model simulation takes about 607.4 s, while the low-fidelity model simulation takes about 48.6 s. Firstly, we can see that HF (70) and FT (50) have similar model construction times of about 14.5 h. Then, the total data generation time was 50.62 and 12.48 h for HF (300) and FT (50), respectively, representing a roughly 75% reduction in data generation time for FT (50). For model training,

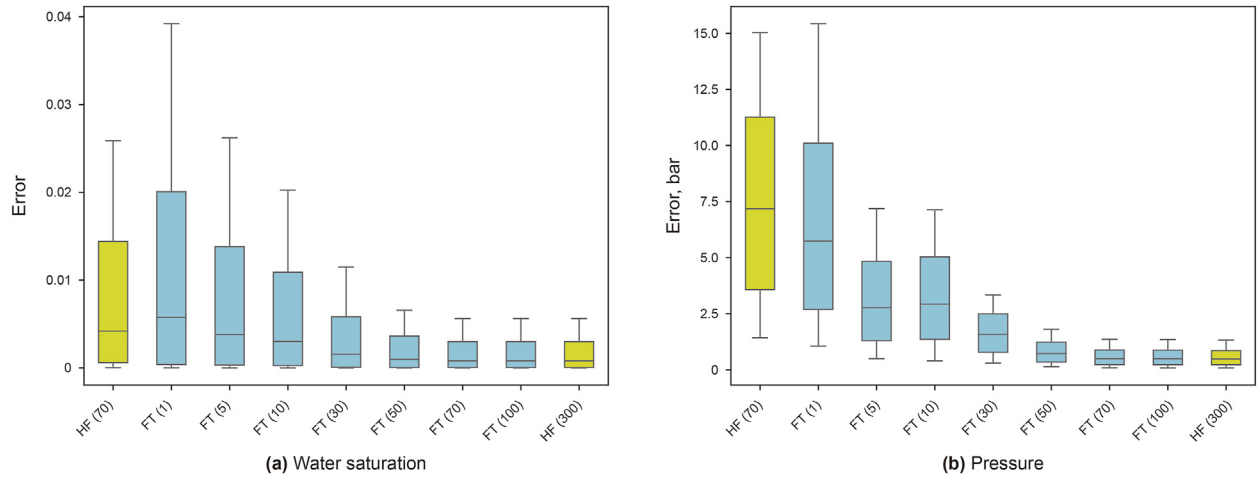


Fig. 18. Box plots of water saturation and pressure prediction errors for different surrogate models. Here HF (N_{hf}) and FT (N_{ft}) stand for models trained using different numbers of high-fidelity and low-fidelity training runs, as defined in Table 4.

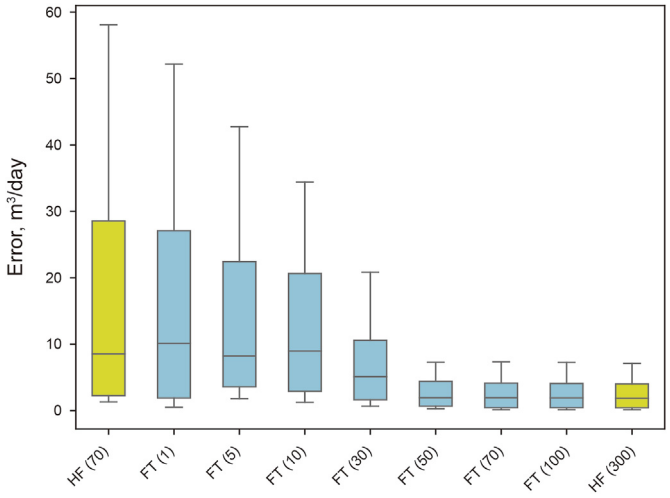


Fig. 19. Box plots of well rate prediction errors for different surrogate models. The colors and notations here have the same meanings as in Fig. 18.

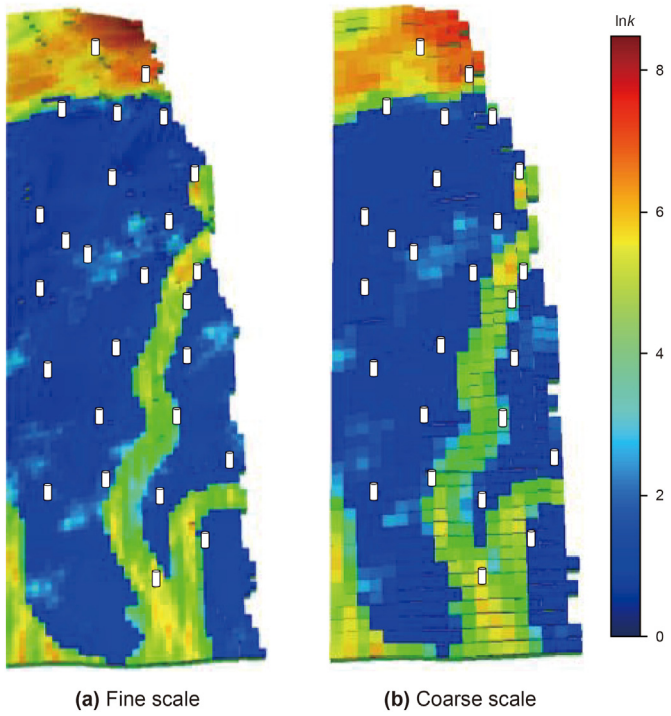


Fig. 20. Distribution of permeability and well locations in the fine-scale and coarse-scale reservoir models.

HF (300) took 10.42 h, while FT (50) took 0.93 h for training on low-fidelity data and 1.22 h for fine-tuning using a smaller number of high-fidelity data. In total, the training of HF (300) and FT (50) surrogate models took 61.04 and 14.63 h, respectively. Therefore, for the case considered here, a total reduction of 76% in computational time can be achieved without sacrificing prediction accuracy using our proposed approach that utilizes computationally inexpensive low-fidelity data for training instead of the entire high-fidelity data.

5. Field application

In this section, we will apply the proposed E-U-Net surrogate model for production optimization of the SAIGUP reservoir (Manzocchi et al., 2008; Matthews et al., 2008). Fig. 20 depicts the distribution of ln$k$ and well location (in white bars) for the fine-scale and corresponding coarse-scale reservoir models. The SAIGUP model has a grid size of 120 × 40 × 20, with 75 m in the x and y directions and 4 m in the z direction. The coarse-scale reservoir model has a grid size of 60 × 20 × 20, with 150 m in the x and y directions and 4 m in the z direction. The reservoir contains both oil and water phases, with 10 water injectors and 16 producers. The initial reservoir pressure was 250 bar. For a detailed description of the SAIGUP reservoir model, please refer to Manzocchi et al. (2008). For the case considered here, the producer BHP range is set between 180 and 230 bar, while the injector BHP range is between 500 and 700 bar. BHP controls in all wells are adjusted every 2 years, for a total of 10 time steps. The surrogate model was trained based on 300 low-fidelity and 50 high-fidelity training runs. For testing, a total of 100 high-fidelity runs, that were not included in the training dataset were used. For the training and test runs, the BHP controls were sampled based on a uniform distribution between the specified ranges for each time step.

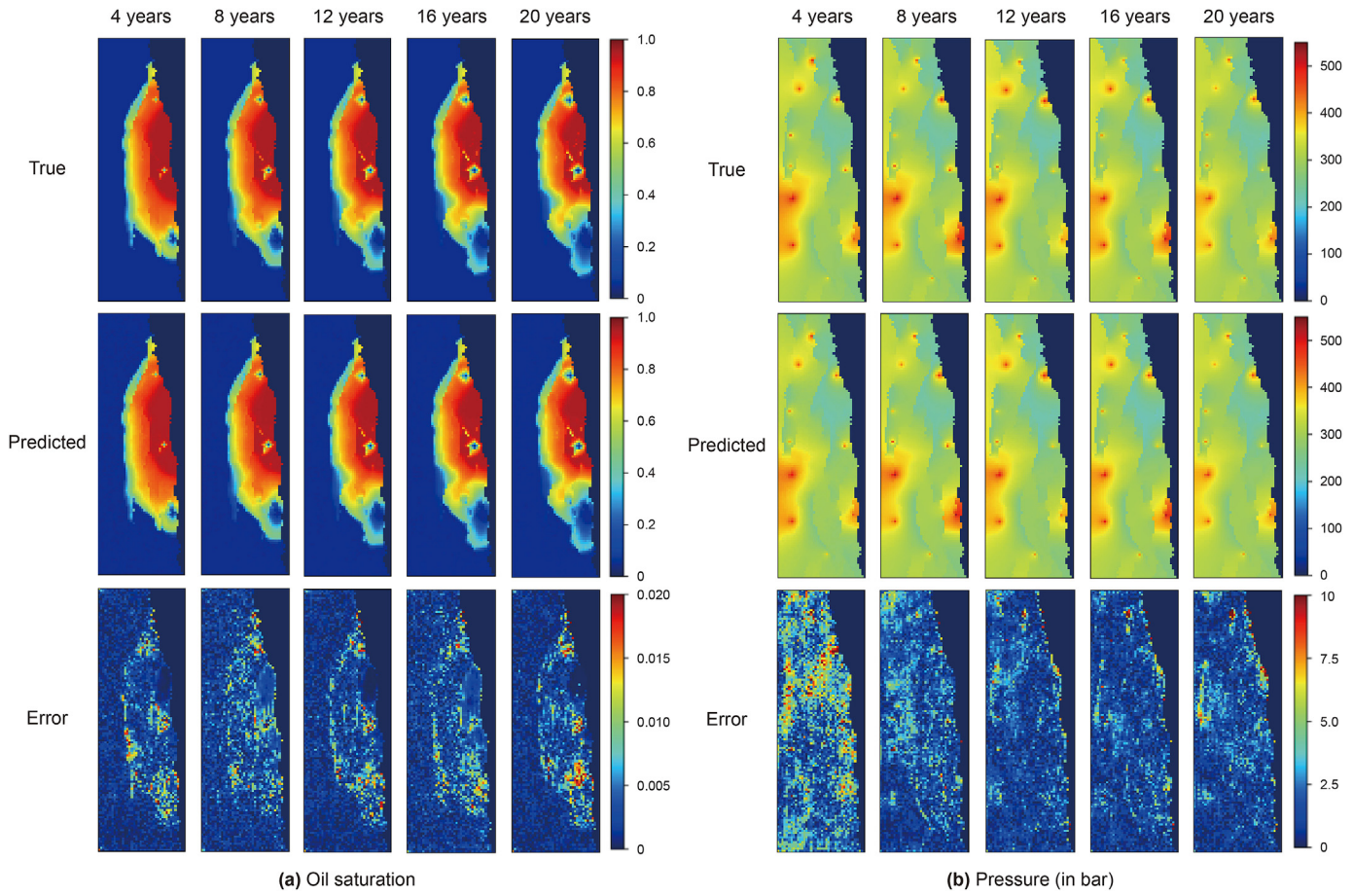


Fig. 21. Oil saturation and pressure prediction distribution in the 20th layer. The first two rows show the results of the reference full-order simulation and the E-U-Net surrogate model, respectively. The third row shows the prediction error of the E-U-Net model. Different columns represent results at different time steps.

Fig. 21 displays the predicted oil saturation and pressure states in the 20th layer at selected time steps for one test run. It is clear that the E-U-Net model provides predictions that are very close to the reference results for both saturation and pressure states at different time steps. The prediction errors for saturation are generally less than 0.02, and the errors for pressure are generally below 5 bar except for a few grid blocks at an early time, as shown in Fig. 21(b). Fig. 22 shows the corresponding well rate prediction. It is evident that the trained E-U-Net model can provide very accurate predictions for well rates (i.e. water and oil production rates for producers and water injection rates for injectors) under a time-varying well control setting.

Fig. 23 compares the predicted cumulative water/oil production and cumulative water injection at each simulation time step for all 100 test runs. The x-axis and y-axis represent the true results and the E-U-Net predicted results. It is evident that the E-U-Net surrogate model provides accurate predictions with small MSE and high R^2 scores (all above 0.99). These results demonstrate the accuracy and robustness of the trained E-U-Net model for the case considered here.

The production optimization process uses the trained E-U-Net surrogate model to replace the time-consuming full-order simulation. During the production optimization process, we aim to identify the time-varying well control variables (BHPs in this case) that maximize the NPV, as calculated in Eq. (9). The economic parameters relevant to the NPV calculation are presented in Table 6.

The E-U-Net based surrogate model developed in this work can be used in various optimization algorithms. Here, it is integrated into the PSO. For a detailed description of PSO, please refer to Table 1. During the production optimization process, the total number of iterations is set to 1000, with 100 particles at each iteration. The BHPs of the producers range from 180 to 230 bar, while the BHPs of the injectors vary from 500 to 700 bar. Each particle represents a potential solution in the 260-dimensional space, corresponding to the control values of the 26 wells over 10 time steps.

Fig. 24 shows the evolution of NPVs during the model training and well-control optimization procedures. The vertical gray dashed lines separate the plot into three sections: NPV calculated from low-fidelity training runs, NPV calculated from high-fidelity runs used for fine-tuning, and optimal NPV for each iteration during PSO optimization. The red markers stand for validation runs at each 100 iterations performed by full-order simulations. There are three key observations here. Firstly, the field development of NPVs can be improved significantly. It is apparent that improved NPVs can be obtained by optimizing the well-control parameters. The optimal NPV from initial full-order high-fidelity simulations is about 4.2×10^9 \$, while the final optimal NPV is about 4.7×10^9 \$. Secondly, the final optimal NPV from the E-U-Net surrogate model prediction is noticeably higher than that from all the training simulations, which demonstrates the extrapolation capability for the trained E-U-Net model. Finally, throughout the iterations, the

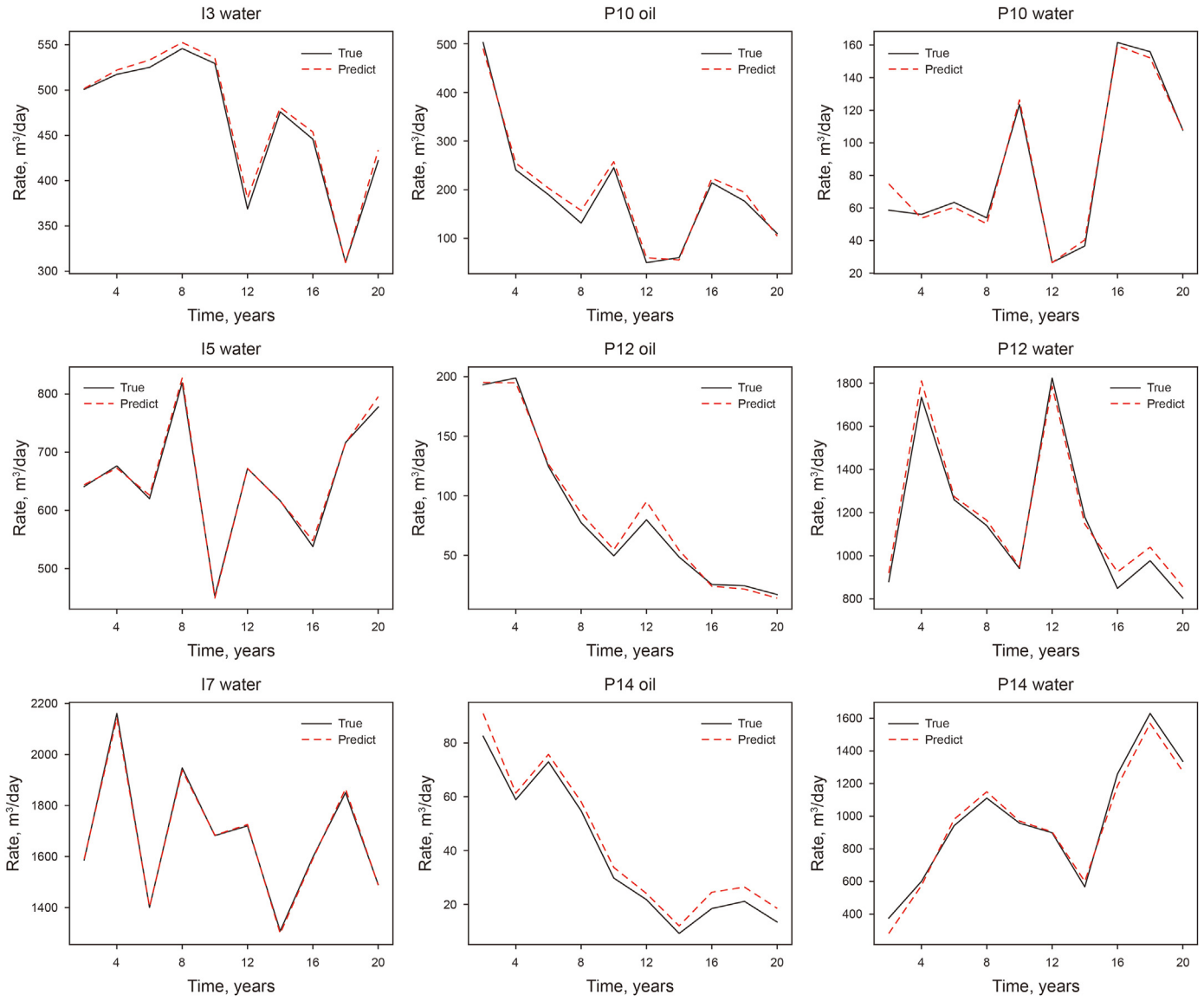


Fig. 22. Well rate prediction at different times. The black line represents true results obtained from full-order simulation, and the red dashed line denotes the predictions from the E-U-Net model.

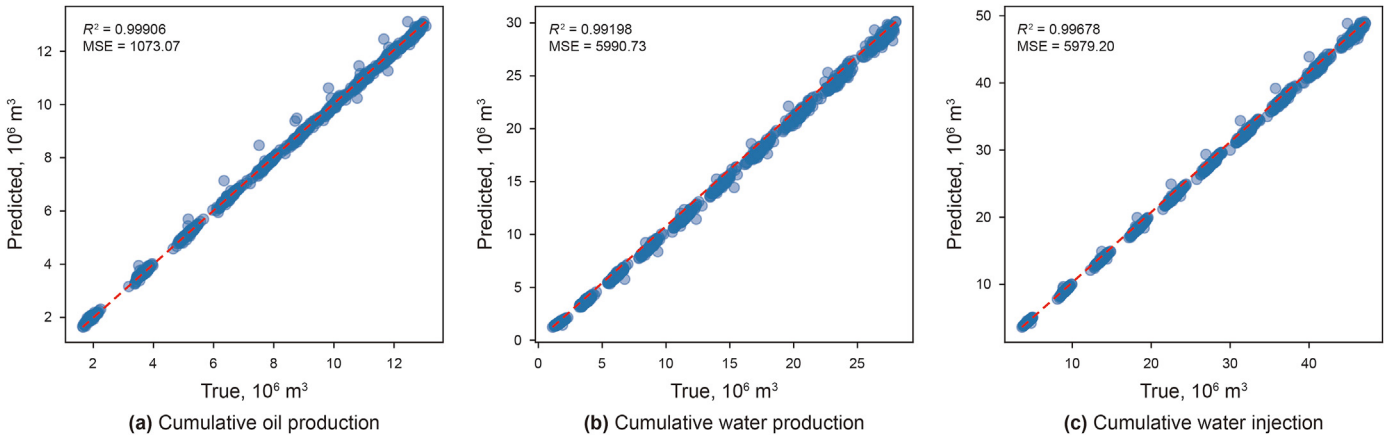


Fig. 23. Crossplots of cumulative oil production, cumulative water production, and cumulative water injection for all 100 test runs at each time step.

Table 6
Economic parameters related to reservoir development NPV calculation.

Economic parameters	Values
Crude oil price r_o, \$/m^3	600
Produced water handling costs c_{pw}, \$/m^3	30
Water injection costs c_{iw}, \$/m^3	30
Producer drilling and completion costs c_o, \$/well	5.0×10^6
Injectors drilling and completion costs c_w, \$/well	5.0×10^6
Average annual discount rate b, %	10.0

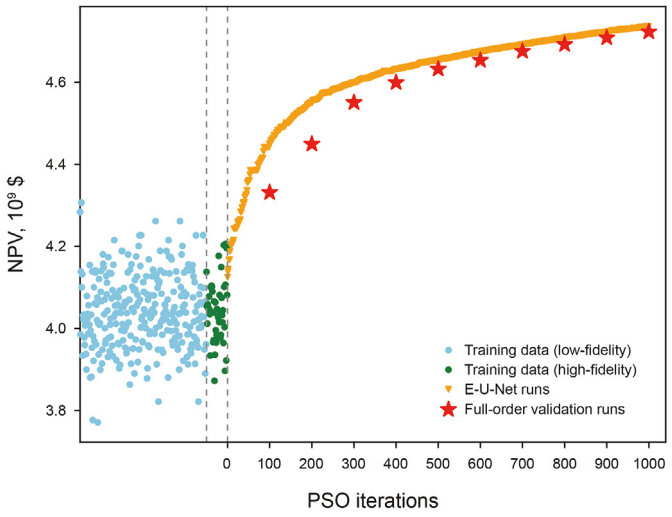


Fig. 24. The evolution of NPV during the optimization process and the NPV of all training runs. Blue and green points stand for the NPV calculated from 300 low-fidelity and 50 high-fidelity training runs, respectively. Yellow triangles are the optimal NPV obtained from the E-U-Net surrogate model at each PSO iteration, and the red stars represent runs from full-order simulations.

trained E-U-Net model remains robust and provides accurate predictions as can be seen from the consistency between E-U-Net and full-order results.

The BHP parameters for the optimal run are displayed in Fig. 25. It is evident that the E-U-Net-based production optimization has resulted in significant variations in the BHP settings of different wells. Some wells have BHP settings that reach the maximum or minimum values within the allowable ranges, such as I9, P3, P12, P14, and P16. The optimization procedure optimized the production parameters for each well according to the specific well productivity and reservoir conditions, which led to improved final NPV as shown in Fig. 24. For example, producer P10 has very high oil rates but low water rates as shown in Fig. 22; therefore, as expected, the corresponding BHP settings were optimized to be at the lower bound, as shown in Fig. 25(b), to enable more fluid to be produced from P10. While producers P12 and P14 have very high water cut, as is apparent in Fig. 22, the final optimized BHP settings were at the upper bound to reduce the associated produced fluids.

Fig. 26 displays the cumulative oil production (COP), cumulative water production (CWP), and cumulative water injection (CWI). The gray curves stand for results from all 100 particles of the first PSO iteration, and the red curves are the final optimized solution. We can see that the optimized solution leads to noticeably high COP, while maintaining slightly higher levels of CWP and CWI. Considering the optimized BHP settings shown in Fig. 25, the results demonstrate the capability of using the E-U-Net surrogate model, in conjunction with PSO optimization algorithms, for reservoir production optimization.

In this section, we used the surrogate model built with multi-fidelity data to optimize production in the SAIGUP reservoir model and achieved satisfactory field-development NPV improvements. The use of the trained E-U-Net surrogate model can significantly reduce the computational cost associated with the optimization process. A total of 300 low-fidelity and 50 high-fidelity simulation runs were used with the E-U-Net model, and the following PSO optimization based on E-U-Net was extremely efficient since each E-U-Net model evaluation only takes a few seconds. Without the surrogate model, a total of 100,000 full-order simulations (1000 iterations multiply 100 runs per iteration) would be required with the standard PSO algorithm used in this study, which can be computationally infeasible.

6. Conclusions

In this study, we developed a novel methodology for constructing a deep-learning-based surrogate model that uses multi-fidelity data in combination with transfer learning. This method aims to accelerate the construction of surrogate models for the

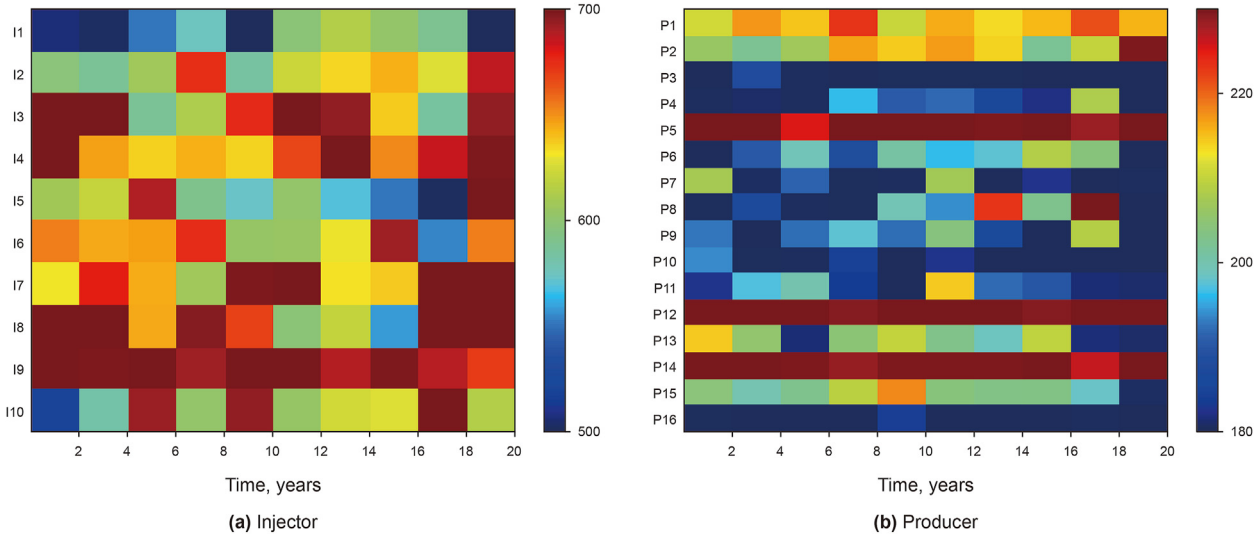


Fig. 25. Optimized BHPs (in bar) of injectors and producers.

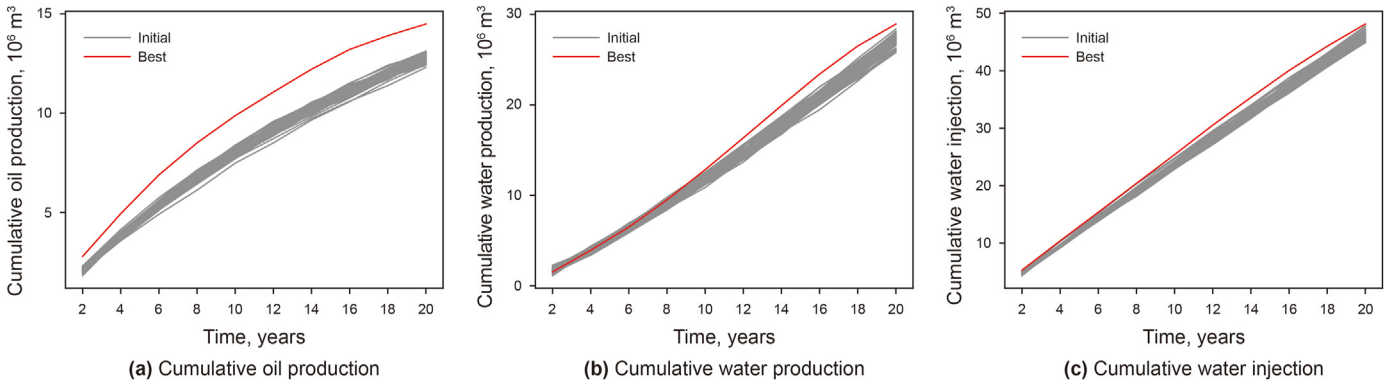


Fig. 26. Initial and final optimized results for cumulative oil production, cumulative water production, and cumulative water injection. Here the initial results correspond to all 100 candidate solutions used for the first iteration of PSO algorithm.

dynamic characterization of subsurface flow under time-varying well controls. The surrogate model is based on an improved 3D U-Net architecture, referred to as E-U-Net, and is trained through a two-step process involving both high-fidelity and low-fidelity training data. The low-fidelity training data is generated efficiently from upscaled reservoir models and used to train a so-called pre-trained E-U-Net model. Parameters of the pre-trained E-U-Net model can then be fine-tuned using a much smaller number of high-fidelity training data. In the E-U-Net model, a dedicated embedding module is also introduced to better capture the effect of well-control input parameters.

A comprehensive analysis of the proposed E-U-Net model was performed through a 3D synthetic model. By testing the surrogate model performance on 100 test cases with randomly sampled well-control settings, the proposed E-U-Net model showed clearly improved predictions of pressure and saturation states than those from the traditional U-Net model. In addition, the simple yet powerful two-step training procedure was shown to provide a significant, ~75%, reduction in total computational cost for surrogate model training, while maintaining the same level of prediction accuracy as the final trained model. Specifically, the prediction median errors for pressure, saturation, and well rate with the surrogate model trained with multi-fidelity data were 0.73 bar, 0.0011, and 1.92 m^3/day, compared to 0.51 bar, 0.0009, and 1.86 m^3/day for the reference surrogate model trained with 300 high-fidelity data. Finally, we applied the proposed surrogate model for production optimization of the SAIGUP reservoir model. The PSO algorithm was used as the optimization engine, though more advanced algorithms can be used. The results showed that the trained E-U-Net surrogate model can provide accurate predictions during the optimization process with an NPV higher than the maximum NPV computed from the training runs, which demonstrated the generalization and extrapolation capabilities of the trained E-U-Net model.

There are several directions to explore for future work. Firstly, the use of a multiple-step (more than two steps as done in this study) training strategy can be explored to further reduce the computational cost of model training. Such improvements will be of interest when training surrogate models for very large-scale reservoir models. Secondly, in this study, the model parameters (i.e.: permeability) are assumed to be known a priori. It will be important to efficiently construct accurate surrogate models considering the existence of uncertain model parameters and time-varying well controls, alleviate the computation burden required for close-loop reservoir management. Finally, we will investigate and improve the proposed approach for more challenging cases, such as three-phase flow and compositional modeling.

CRediT authorship contribution statement

Jia-Wei Cui: Writing – original draft, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Wen-Yue Sun:** Writing – review & editing, Visualization, Validation, Supervision, Project administration, Methodology, Funding acquisition, Data curation, Conceptualization. **Hoonyoung Jeong:** Writing – review & editing, Funding acquisition. **Jun-Rong Liu:** Writing – review & editing, Funding acquisition. **Wen-Xin Zhou:** Writing – review & editing.

Data availability

The code and data that support the findings of this study will be made available on GitHub (https://github.com/Jiawei0106/3D-E-U-Net) upon acceptance of the manuscript for publication.

Conflict of interest statement

No conflict of interest exists in the submission of this manuscript, and the manuscript is approved by all authors for publication. I would like to declare on behalf of my co-authors that the work described is original research that has not been published previously, and is not under consideration for publication elsewhere, in whole or in part. All the authors listed have approved the manuscript that is enclosed.

Acknowledgments

We are grateful to the primary funding support from the National Natural Science Foundation of China (No. 52204065, No. ZX20230398). Hoonyoung Jeong was supported by a grant from the Human Resources Development Program (No. 20216110100070) of the Korea Institute of Energy Technology Evaluation and Planning (KETEP).

References

Afshar, M.H., 2013. Extension of the constrained particle swarm optimization algorithm to optimal operation of multi-reservoirs system. Int. J. Electr. Power Energy Syst. 51, 71–81. https://doi.org/10.1016/j.ijepes.2013.02.035.

An, Z., Zhou, K., Hou, J., et al., 2022. Accelerating reservoir production optimization by combining reservoir engineering method with particle swarm optimization algorithm. J. Petrol. Sci. Eng. 208, 109692. https://doi.org/10.1016/j.petrol.2021.109692.

Aziz, K., 1999. Fundamentals of Reservoir Simulation. Stanford University, pp. 30–50.

Chen, Y., Mallison, B.T., Durlofsky, L.J., 2008. Nonlinear two-point flux approximation for modeling full-tensor effects in subsurface flow simulations. Comput.

Geosci. 12 (3), 317–335. https://doi.org/10.1007/s10596-007-9067-5.
Christie, M.A., 1996. Upscaling for reservoir simulation. J. Petrol. Technol. 48 (11), 1004–1010. https://doi.org/10.2118/37324-JPT.
De, S., Britton, J., Reynolds, M., et al., 2020. On transfer learning of neural networks using bi-fidelity data for uncertainty propagation. Int. J. Uncertain. Quantification 10 (6). https://doi.org/10.1615/int.j.uncertaintyquantification.2020033267.
Du, S., Zhao, X., Xie, C., et al., 2023. Data-driven production optimization using particle swarm algorithm based on the ensemble-learning proxy model. Pet. Sci. 20 (5), 2951–2966. https://doi.org/10.1016/j.petsci.2023.04.001.
Geneva, N., Zabaras, N., 2020. Multi-fidelity generative deep learning turbulent flows. Foundations of Data Science 2 (4), 391–428. https://doi.org/10.1016/j.petsci.2023.04.001.
He, J., Sætrom, J., Durlofsky, L.J., 2011. Enhanced linearized reduced-order models for subsurface flow simulation. J. Comput. Phys. 230 (23), 8313–8341. https://doi.org/10.1016/j.jcp.2011.06.007.
Huang, H., Gong, B., Sun, W., et al., 2023. Application of an improved deep-learning framework for large-scale subsurface flow problems with varying well controls. SPE J. 29 (1), 574–591. https://doi.org/10.2118/217456-PA.
Jiang, S., Durlofsky, L.J., 2023. Use of multi-fidelity training data and transfer learning for efficient construction of subsurface flow surrogate models. J. Comput. Phys. 474, 111800. https://doi.org/10.1016/j.jcp.2022.111800.
Jin, L., Lu, H., Wen, G., 2019. Fast uncertainty quantification of reservoir simulation with variational U-Net. arXiv. https://doi.org/10.48550/arXiv.1907.00718.
Jin, Z.L., Liu, Y., Durlofsky, L.J., 2020. Deep-learning-based surrogate model for reservoir simulation with time-varying well controls. J. Petrol. Sci. Eng. 192, 107273. https://doi.org/10.1016/j.petrol.2020.107273.
Kim, J., Park, C., Ahn, S., et al., 2021. Iterative learning-based many-objective history matching using deep neural network with stacked autoencoder. Pet. Sci. 18 (5), 1465–1482. https://doi.org/10.1016/j.petsci.2021.08.001.
Kim, Y.D., Durlofsky, L.J., 2021. A recurrent neural network-based proxy model for well-control optimization with nonlinear output constraints. SPE J. 26 (4), 1837–1857. https://doi.org/10.2118/203980-PA.
Kim, Y.D., Durlofsky, L.J., 2023. Convolutional-recurrent neural network proxy for robust optimization and closed-loop reservoir management. Comput. Geosci. 27 (2), 179–202. https://doi.org/10.1007/s10596-022-10189-9.
Kingma, D.P., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. https://doi.org/10.48550/arXiv.1412.6980.
Kochenderfer, M.J., 2019. Algorithms for Optimization. The MIT Press Cambridge, pp. 158–159.
Kumar, N., Mohapatra, S.K., Ragit, S.S., et al., 2017. Optimization of safflower oil transesterification using the Taguchi approach. Pet. Sci. 14, 798–805. https://doi.org/10.1007/s12182-017-0183-0.
Li, H., Chen, Y., Rojas, D., et al., 2014. Development and application of near-well multiphase upscaling for forecasting of heavy oil primary production. J. Petrol. Sci. Eng. 113, 81–96. https://doi.org/10.1016/j.petrol.2014.01.002.
Li, H., Durlofsky, L.J., 2016. Ensemble level upscaling for compositional flow simulation. Comput. Geosci. 20, 525–540. https://doi.org/10.1007/s10596-015-9503-x.
Lin, M., 2013. Network in network. arXiv preprint arXiv:1312.4400. https://doi.org/10.48550/arXiv.1312.4400.
Manzocchi, T., Matthews, J.D., Strand, J.A., et al., 2008. A study of the structural controls on oil recovery from shallow-marine reservoirs. Pet. Geosci. 14 (1), 55–70. https://doi.org/10.1144/1354-079307-786.
Matthews, J.D., Carter, J.N., Stephen, K.D., et al., 2008. Assessing the effect of geological uncertainty on recovery estimates in shallow-marine reservoirs: The application of reservoir engineering to the SAIGUP project. Pet. Geosci. 14 (1), 35–44. https://doi.org/10.1144/1354-079307-791.
Meng, X., Karniadakis, G.E., 2020. A composite neural network that learns from multi-fidelity data: Aplication to function approximation and inverse PDE problems. J. Comput. Phys. 401, 109020. https://doi.org/10.1016/j.jcp.2019.109020.
Mo, S., Zabaras, N., Shi, X., et al., 2020. Integration of adversarial autoencoders with residual dense convolutional networks for estimation of non-Gaussian hydraulic conductivities. Water Resour. Res. 56 (2), e2019WR026082. https://doi.org/10.1029/2019WR026082.
Müller, S., Schüler, L., Zech, A., et al., 2020. GSTools v1. 3: A toolbox for geostatistical modelling in Python. Geosci. Model Dev. (GMD) 15 (7), 3161–3182. https://doi.org/10.5194/gmd-15-3161-2022.
Nwachukwu, A., Jeong, H., Pyrcz, M., et al., 2018. Fast evaluation of well placements in heterogeneous reservoir models using machine learning. J. Petrol. Sci. Eng. 163, 463–475. https://doi.org/10.1016/J.PETROL.2018.01.019.
Pan, S., Yang, Q., 2009. A survey on transfer learning. IEEE Trans. Knowl. Data Eng. 22 (10), 1345–1359. https://doi.org/10.1109/TKDE.2009.191.
Poli, R., Kennedy, J., Blackwell, T., 2007. Particle swarm optimization. Swarm Intelligence 1 (1), 33–57. https://doi.org/10.1007/s11721-007-0002-0.
Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional Networks for Biomedical Image Segmentation. Springer International Publishing, Cham, pp. 234–241. https://doi.org/10.1007/978-3-319-24574-4_28.
Shang, J., Ji, Z., Qiu, M., et al., 2019. Multi-objective optimization of high-sulfur natural gas purification plant. Pet. Sci. 16, 1430–1441. https://doi.org/10.1007/s12182-019-00391-3.
Song, D.H., Tartakovsky, D.M., 2022. Transfer learning on multi-fidelity data. Journal of Machine Learning for Modeling and Computing 3 (1). https://doi.org/10.1615/JMachLearnModelComput.2021038925.
Sun, A.Y., 2020. Optimal carbon storage reservoir management through deep reinforcement learning. Appl. Energy 278, 115660. https://doi.org/10.1016/j.apenergy.2020.115660.
Sun, W., Hui, M.H., Durlofsky, L.J., 2017. Production forecasting and uncertainty quantification for naturally fractured reservoirs using a new data-space inversion procedure. Comput. Geosci. 21, 1443–1458. https://doi.org/10.1007/s10596-017-9633-4.
Tang, M., Liu, Y., Durlofsky, L.J., 2020. A deep-learning-based surrogate model for data assimilation in dynamic subsurface flow problems. J. Comput. Phys. 413, 109456. https://doi.org/10.1016/j.jcp.2020.109456.
Tang, M., Liu, Y., Durlofsky, L.J., 2021. Deep-learning-based surrogate flow modeling and geological parameterization for data assimilation in 3D subsurface flow. Comput. Methods Appl. Mech. Eng. 376, 113636. https://doi.org/10.1016/j.cma.2020.113636.
Wang, J., Zhang, L., Zhang, K., et al., 2024. Multi-surrogate framework with an adaptive selection mechanism for production optimization. Pet. Sci. 21 (1), 366–383. https://doi.org/10.1016/j.petsci.2023.08.028.
Xu, J., Fu, Q., Li, H., 2023. A novel deep learning-based automatic search workflow for CO_2 sequestration surrogate flow models. Fuel 354, 129353. https://doi.org/10.1016/j.fuel.2023.129353.
Zhang, K., Wang, Y., Li, G., et al., 2021. Prediction of field saturations using a fully convolutional network surrogate. SPE J. 26 (4), 1824–1836. https://doi.org/10.2118/205485-PA.
Zhang, K., Yu, H., Ma, X., et al., 2022. Multi-source information fused generative adversarial network model and data assimilation based history matching for reservoir with complex geologies. Pet. Sci. 19 (2), 707–719. https://doi.org/10.1016/j.petsci.2021.10.007.
Zhang, Y., Wang, H., Che, J., et al., 2021. Multi-objective optimization and experiment of nylon cord rubber in expandable packer. Pet. Sci. 18, 269–284. https://doi.org/10.1007/s12182-020-00539-6.
Zhong, Z., Sun, A.Y., Ren, B., et al., 2021. A deep-learning-based approach for reservoir production forecast under uncertainty. SPE J. 26 (3), 1314–1340. https://doi.org/10.2118/205000-PA.
Zhu, Y., Zabaras, N., 2018. Bayesian deep convolutional encoder-decoder networks for surrogate modeling and uncertainty quantification. J. Comput. Phys. 366, 415–447. https://doi.org/10.1016/j.jcp.2018.04.018.