Efficient simulation of CO_2 migration dynamics in deep saline aquifers using a multi-task deep learning technique with consistency

Zhao, Mengjie; Wang, Yuhang; Gerritsma, Marc; Hajibeygi, Hadi

DOI
10.1016/j.advwatres.2023.104494

Publication date
2023

Document Version
Final published version

Published in
Advances in Water Resources

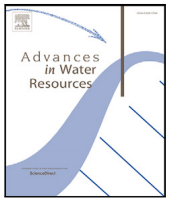
Citation (APA)
Zhao, M., Wang, Y., Gerritsma, M., & Hajibeygi, H. (2023). Efficient simulation of CO_2 migration dynamics in deep saline aquifers using a multi-task deep learning technique with consistency. *Advances in Water Resources*, *178*, Article 104494. https://doi.org/10.1016/j.advwatres.2023.104494

Advances in Water Resources 178 (2023) 104494

Contents lists available at ScienceDirect

Advances in Water Resources

journal homepage: www.elsevier.com/locate/advwatres



Efficient simulation of CO_2 migration dynamics in deep saline aquifers using a multi-task deep learning technique with consistency

Mengjie Zhao [a,*], Yuhang Wang [b], Marc Gerritsma [a], Hadi Hajibeygi [c]

[a] Faculty of Aerospace Engineering, Department of Flow Physics and Technology, Delft University of Technology, Kluyverweg 2, 2629 HS, Delft, The Netherlands
[b] School of Environmental Studies, China University of Geosciences, Wuhan 430078, China
[c] Faculty of Civil Engineering and Geosciences, Department of Geoscience and Engineering, Delft University of Technology, Stevinweg 1, 2628CV, Delft, The Netherlands

ARTICLE INFO

Keywords:
Geologic carbon storage
Multiphase flow in porous media
Deep learning method
Multi-task learning

ABSTRACT

CO_2 sequestration and storage in deep saline aquifers is a promising technology for mitigating the excessive concentration of the greenhouse gas in the atmosphere. However, accurately predicting the migration of CO_2 plumes requires complex multi-physics-based numerical simulation approaches, which are prohibitively expensive due to highly nonlinear coupled governing equations and uncertainties in heterogeneous spatial parameter distributions. To address this challenge, we developed an end-to-end deep learning workflow employing encoder–decoder architectures with residual network (ResNet) to efficiently predicts the spatial–temporal evolution of the solution CO_2-brine ratio (R_s) and gas saturation (S_g) – the two essential tasks for quantifying the amount of trapped CO_2 – given heterogeneous permeability fields as input. Specifically, we introduce a general multi-task learning with consistency (MTLC) framework to simultaneously predict R_s and S_g. The MTLC model leverages related tasks with less computational expensive labeled datasets to improve generalization ability. In our study, predictions for multiple tasks from the same permeability realization are not independent but expected to be consistent, as the proposed framework utilizes data-driven cross-task consistency constraints to augment learning of related tasks. Our deep learning model is trained based on physical trapping mechanisms, which play a dominant role in the CO_2 migration process. The results demonstrate that the MTLC model with joint learning yields more accurate predictions and improved generalization for predicting CO_2 migration in several test cases. Furthermore, our workflow is 10^5 times faster than a high-fidelity physics-based numerical simulator, making it a viable alternative for field-scale applications.

1. Introduction

Carbon capture and storage (CCS) is a viable and effective strategy for reducing anthropogenic CO_2 emissions into the atmosphere, which is beneficial to mitigate climate change (Kopp et al., 2010; Selma et al., 2014; Liu et al., 2020; Krevor et al., 2023). In CCS, the captured CO_2 is compressed into the super-critical fluid status, and then injected into the geological medium such as depleted oil and gas reservoirs and deep saline aquifers for long-term sequestration (Celia et al., 2015). Interest in CO_2 storage in saline aquifers has grown recently due to the large storage capacity under safe operational conditions (Wang et al., 2022). For storing in saline aquifers, the CO_2 migration process can be described by the physics of multicomponent, multiphase flow and transport in porous media. To date, high-fidelity numerical simulation of fully physics-coupled flow is the primary tool to reliably monitor the CO_2 plume evolution and migration because of various trapping mechanisms for the sake of security (Ide et al., 2007). Despite some success in numerical simulation strategies, the computational expense of solving the highly nonlinear discretized mass conservation equations of brine and CO_2 is prohibitively high in practical applications of uncertainty quantification and history matching, which require a large number of forward simulation runs (Tang et al., 2022; Wang et al., 2023). Due to the inherent uncertainty of the spatial distributions of rock properties and the complexity and heterogeneties of the geological formation structures, improved dynamic migration forecasting capability is important for effective CCS projects (Elenius et al., 2015).

In recent years, the significance of surrogate models has grown with the expansion of the machine learning field, providing a more computationally efficient alternative to physics-based numerical simulation (Guo and Reynolds, 2018; Zhao et al., 2020b; Kamrava et al., 2021). While various machine learning techniques have been employed

* Corresponding author.
E-mail addresses: m.zhao-2@tudelft.nl (M. Zhao), wangyuhang17@cug.edu.cn (Y. Wang), m.i.gerritsma@tudelft.nl (M. Gerritsma), h.hajibeygi@tudelft.nl (H. Hajibeygi).

https://doi.org/10.1016/j.advwatres.2023.104494
Received 6 March 2023; Received in revised form 6 June 2023; Accepted 19 June 2023
Available online 21 June 2023

to construct surrogate models for multiphase flow, such as Gaussian Process (Hamdi et al., 2017), and Radial Basis Function (RBF) (Chen et al., 2020b; Babaei and Pan, 2016), their accuracy tends to be limited by the curse of dimensionality (Chen et al., 2020a). For example, GP is generally effective only for applications with up to 20 parameters (Zhao et al., 2020a). While combining GP with dimensionality reduction methods has shown promise for certain prediction tasks, such techniques can lead to a loss of information, potentially affecting the overall performance of the surrogate model. Obviously, some surrogate models with efficient strategies have been successfully applied in specific contexts, such as the combination of a RBF model and proper sampling strategy significantly reduce the computational cost associated with coastal aquifer management while maintaining the accuracy (Christelis et al., 2018). Nevertheless, the problem in our work demands stronger capabilities from surrogate models, making deep learning (DL) techniques an appropriate choice. DL techniques have demonstrated their potential for handing high-dimensional regression problems and are widely applied in computer vision (CV) tasks like object detection and image segmentation with the rapid development of graphics processing units (GPU) (Redmon et al., 2016; Ren et al., 2015). Considering this, our goal in this paper is to develop a novel deep learning-based surrogate model capable of effectively capturing the spatio-temporal evolution of CO_2 plume migration, thus facilitating the management of CCS operations.

In the general context of the fluid flow and transport in porous media, there has been some promising research using deep neural networks to predict mappings from the reservoir properties (e.g. permeability) to the state variables (e.g. pressure or saturation maps) (Wang et al., 2021). Neural networks rely on the simulation data to train a statistical model to approximate the input–output relationship of interests and can theoretically estimate any complex function given adequate setup and training. To be specific, a convolutional neural network (CNN) is a powerful algorithm for image processing which can capture spatial features while reducing the number of free parameters, based on the shared-weight architecture of the convolutional layers or filters (Simonyan and Zisserman, 2014). Zhu and Zabaras (2018) first employed a fully convolutional encoder–decoder network to approximate the single-phase steady state flow in geological models characterized by Gaussian permeability fields, and demonstrated that their neural network, trained with a limited amount of data, was able to predict high-dimensional pressure maps. Then, Tang et al. (2020) developed a Recurrent residual U-Net for data assimilation in dynamic subsurface flow problems, in which a residual U-Net and a recurrent architecture Long short-term memory (LSTM) are combined to capture both spatial and temporal information. Kadeethum et al. (2021) used generative adversarial network (GAN) based deep learning model for predicting the solution of steady state in heterogeneous porous media. Some similar network architectures are applied to CO_2 storage problems. Wen et al. (2021a,b) extend the U-Net to translate permeability and injection parameters to CO_2 saturation maps. Zhong et al. (2019) incorporated U-Net with GAN to capture the time dependence. Besides, Yan et al. (2022) applied Fourier Neural Operators (FNO) to predict the temporal–spatial evolution of CO_2 plumes during injection and post-injection periods, respectively. Chu et al. (2023) presented deep learning-based surrogate models with four different algorithms and a physics-framed two-phase flow problem involving displacement of water by CO_2. They compared the Multi-layer perception (MLP), CNN, LSTM cell, and gated recurrent unit (GRU) cell, and the MLP demonstrated good performance with most trainable parameters.

However, previous studies on CO_2 storage primarily adopted the single-task learning approaches, focusing on one of the state variables without considering the consistency among the variables, i.e., perform either pressure or saturation exclusively. While these studies also achieved the good performance through deep learning techniques, they may not capture the complex interrelationships between difference aspects of the CO_2 storage process, since most variables are connected by the underlying physics observed. Furthermore, single-task learning methods often require a large amount of labeled data. When the reservoir realization and well control parameters are given, the dynamic systems are determined by the forward simulation using a finite volume method. Suppose a CO_2 plume migration process is simulated, the prediction of the amount of CO_2 in the liquid phase and in the gas phase are dependent, and consequently should enforce consistency constraints on each other (Zamir et al., 2020). Therefore, in contrast to the single-task methods, joint-task learning methods have shown promise in improving predictions by utilizing task-correlative information and sharing informative features between related tasks (Liu et al., 2019; Zhang et al., 2019). Additionally, a single model performing multiple tasks jointly can be more efficient than using separate and independent models for each task. This type of deep learning is named Multi-task Leaning (MTL).

Inspired by the preceding ideas, we present a novel deep learning surrogate framework for CO_2 storage in deep saline aquifers which integrates multi-task learning with consistency constraints (MTLC). In this work, the workflow is based on encoder–decoder architectures consisting of ResNet blocks, and is designed to simultaneously predict CO_2-brine ratio (R_s) and gas saturation (S_g), the two essential tasks for quantifying trapped CO_2, given heterogeneous permeability fields as input. Then we demonstrate the performance in training surrogate model for predicting the dynamic R_s and S_g maps simultaneously in layered formations. The MTLC model leverages related tasks with fewer computationally expensive labeled datasets to improve generalization ability and utilizes data-driven cross-task consistency constraints to augment learning of related tasks. It offers a more efficient and accurate approach to understanding and managing geological CO_2 sequestration. The rest of this paper is organized as follows. In Section 2, we introduce a general underlying physics and define the problem of interests. Then, the framework of MTLC for CO_2 storage is described and used to predict in Sections 3 and 4. Finally, conclusions are provided in the last section.

2. Problem definition

2.1. Governing equations

For the CO_2 and brine multi-component multi-phase flow problem, the general form of mass conservation equation is described as (Wang et al., 2022):

$$\frac{\partial}{\partial t}(\phi \sum_{\alpha} x_{c,\alpha}\, \rho_\alpha S_\alpha) + \nabla \cdot (\sum_{\alpha} x_{c,\alpha}\, \rho_\alpha \boldsymbol{u}_\alpha) - \sum_{\alpha} x_{c,\alpha}\, q_\alpha = 0, \tag{2.1}$$

where, the first term is the fluid accumulation, the second is the advective term, and the third is the source or sink term. Subscript α and c indicate phases (l: liquid and g: gas) and components (CO_2 and brine), respectively. ϕ is the rock porosity; t is time; S_α, ρ_α and q_α are saturation, density and source term of phase α; $x_{c,\alpha}$ represents the molar fraction of component c in phase α; $\boldsymbol{u}_\alpha$ is the Darcy velocity given by:

$$\boldsymbol{u}_\alpha = -\frac{k k_{r\alpha}}{\mu_\alpha} \nabla \cdot (P_\alpha - \rho_\alpha g h), \tag{2.2}$$

where, k is the absolute permeability of the rock, $k_{r\alpha}$, μ_α and P_α denote relative permeability, viscosity and pressure of phase α, respectively. g is the gravitational acceleration, h is the depth with respect to a set reference. The phase pressures are related to each other through the capillary pressure P_c.

According to the overall composition variable set, the Eq. (2.1) can be expressed as:

$$\frac{\partial}{\partial t}(\phi \rho_T z_c) + \nabla \cdot (\sum_{\alpha} x_{c,\alpha}\, \rho_\alpha \boldsymbol{u}_\alpha) - \sum_{\alpha} x_{c,\alpha}\, \rho_\alpha q_\alpha = 0, \tag{2.3}$$

where, ρ_T is the total density and z_c is the mole fraction. Finally, the system is closed by enforcing $S_g + S_l = 1$. In a fully-physics-based reservoir simulator, these equations are solved iteratively with a

fully implicit scheme to calculate the primary variables of liquid phase pressure (P_l) and CO_2 mole fraction (z_{CO_2}).

Trapping mechanisms for retraining CO_2 in deep saline aquifers have been categorized into four types (Rubin and De Coninck, 2005): (a) Structural and stratigraphic trapping. This refers to the upward migration is inhibited by the overlying low-permeable cap rock. (b) Residual trapping. This occurs because CO_2 is typically the non-wetting phase in many sedimentary rocks. As a result, brine, the wetting phase, tends to imbibe into the trailing edge of the CO2 plume. (c) Dissolution trapping. This takes place when CO_2 comes into contact with undersaturated brine, dissolving into it. (d) Mineral trapping. This happens when dissolved CO_2 reacts with the reservoir rock's minerals. Saline aquifers are often found in sandstone, a siliciclastic rock with a high quartz content. This mineral has an insignificant contribution to geochemical reactions during the trapping process (Gunter et al., 2000). As a result, our subsequent analysis will focus on the first three physical trapping mechanisms, which are collectively referred to as hydrodynamic trapping. Besides, to simplify the problem setting, our simulation does not explicitly include molecular diffusion and hydrodynamic dispersion, which is a common assumption in CO_2 storage simulation. Besides, we assume that CO_2 from the gas phase can dissolve in the liquid phase, but dissolution of brine in the gas phase is neglected (i.e. the gas phase contains only one component).

2.2. Surrogate modeling for CCS

The above governing equations are numerically solved by the fully-physics simulator, DARSim. DARSim uses the finite volume method (FVM) with fully implicit scheme for simulation (Cusini et al., 2016). We are concerned with the dynamics of interaction between CO_2 and brine, which is a challenging problem due to the complex interplay between viscous, capillary, and gravitational forces.

Moreover, in order to quantify the impact of different trapping mechanisms and uncertain reservoir properties, we design heterogeneous permeability realizations, and then calculate the amount of CO_2 trapped by dissolution and residual trapping, which are considered to be secure in hydrodynamic trapping. They are calculated based on the solution CO_2-brine ratio R_s and gas phase saturation maps S_g, respectively. In particular, R_s is described as:

$$R_s = \frac{\rho^{\mathrm{STC}}_{\mathrm{brine}} x_{\mathrm{CO}_2,l}}{\rho^{\mathrm{STC}}_{\mathrm{CO}_2}(1 - x_{\mathrm{CO}_2,l})}, \tag{2.4}$$

where, the superscript 'STC' represents the property at standard conditions.

However, we need to solve the discretized versions of Eq. (2.3) hundreds or thousands of times for uncertainty quantification. Suppose the computation simulation is considered as a black-box mapping, a single simulation run can be described as:

$$\mathbf{y} = f(\mathbf{x}), \tag{2.5}$$

where, $\mathbf{x} \in \mathbb{R}^{n_s}$ denotes the high-dimensional realization of random field, n_s is the total number of grid blocks, f indicates the simulation-induced function, $\mathbf{y} \in \mathbb{R}^{2n_s n_t}$ is the dynamic response maps (R_s and S_g) at n_t time steps.

In order to resemble an inexpensive replacement of the numerical simulator, deep learning models with the $\mathbf{y} \approx \hat{\mathbf{y}} = \mathcal{F}(\mathbf{x}, \theta)$ are trained with a limited simulation dataset $\{(\mathbf{x}^i, \mathbf{y}^i)\}_{i=1}^{N}$ to approximate the relationship between the input properties $\mathbf{x}$ and the corresponding multiple dynamic responses $\mathbf{y}$, where θ are the deep neural network parameters, and N is the number of training simulation-based data. Therefore, the main task is transformed to an image-to-image regression problem which requires pixel-wise predictions as $\mathcal{F} : \mathbb{R}^{n_s} \rightarrow \mathbb{R}^{2n_s n_t}$. Therefore, our goal in this work is to develop a surrogate model to provide the time-dependent states $\hat{\mathbf{y}}$ given a permeability map $\mathbf{x}$.

3. Methodology

3.1. Single-task Leaning (STL)

3.1.1. Deep encoder–decoder architecture for image-to-image regression

Given the input database $(\mathbf{x}, \mathbf{y})$, typical deep neural networks approximate the input–output relationship $\hat{f} : \mathbf{x} \rightarrow \mathbf{y}$ through a number of fully connected layers which is described as:

$$\mathbf{a}^l = h^l(\mathbf{a}^{l-1}) = \sigma^l(\mathbf{W}^l \mathbf{a}^{l-1} + \mathbf{b}^l), \tag{3.1}$$

where, $\mathbf{W}^l$ and $\mathbf{b}^l$ are the weights and biases for the lth layer with input $\mathbf{a}^{l-1}$ and output $\mathbf{a}^l$, respectively. σ^l denote the nonlinear activation function. Therefore, the approximate function is $\hat{f} = h^L \cdot h^{L-1} \cdots h^1$, where L is the depth of the network.

However, a fully connected neural network leads to an extremely large number of trainable parameters when dealing with the high-dimensional problems. CNNs are commonly applied to reduce the number of parameters greatly due to the parameter sharing scheme. Therefore, they are widely used for image processing and are able to extract the feature of inputs (Albawi et al., 2017). A convolutional layer consists of a series of convolution kernels which are used to compute the feature maps that are essentially matrices. Suppose that we have some $S \times S$ square neuron layer which is followed by convolutional layer. If we use an $m \times m$ filter $\boldsymbol{\omega}$, the output will be of size $(S-m+1) \times (S-m+1)$, and the feature value $h_{i,j}(x_{i,j})$ at location (i, j) is the sum of contributions (weighted by the filter components):

$$h_{i,j}(x_{i,j}) = \sigma\Big(\sum_{k_i=0}^{m-1} \sum_{k_j=0}^{m-1} \omega_{k_i,k_j} x_{i+k_i,j+k_j}\Big), \tag{3.2}$$

Therefore, the feature maps of a convolutional layer consisting of N_k filters are $\{\boldsymbol{h}^l, l = 1, \ldots, N_k\}$.

In contrast to traditional CNNs, which consist of a series of convolutional layers followed by fully connected layers, a popular model design pattern for pixel-wise predictions is the encoder–decoder architecture. It replaces the last fully-connected layers with upsampling or deconvolution layers to recover resolution, exhibiting promising performance in handling mappings between high-dimensional inputs and outputs (Mo et al., 2019). The encoder–decoder neural network employs a coarse-refine process, where the encoder reduces spatial dimensions in every layer and increases channels to extract higher-level features at lower spatial resolution, while the decoder increases spatial dimensions and reduces channels to refine the image representation and construct the output. Ultimately, the spatial dimensions are restored to make predictions for each input image pixel (Badrinarayanan et al., 2017). This type of model has been successfully utilized in various fields, including computer vision for image segmentation (Ronneberger et al., 2015). Moreover, there are also successful applications in the general context of multiphase flow (Zhu and Zabaras, 2018; Zhu et al., 2019). Therefore, a fully convolutional encoder–decoder architecture is employed to formulate our approach.

3.2. Residual Neural Network (ResNet)

When dealing with deep CNNs to solve a complicated task, the general operation is to engage in stacking more layers. These additional layers help solve complex problems more efficiently as the different layers could be trained for varying tasks to get highly accurate results. While the number of stacked layers can enrich the feature of the model, a deeper network can show the issue of degradation. In other words, as the number of layers of the neural network increases, the accuracy levels may get saturated and slowly degrade after some points. As a result, the performance of the model deterioates both on the training and testing datasets. This degradation is the results of the problem of vanishing or exploding gradients (He et al., 2016a).

In order to solve this problem, the ResNet architecture is introduced with the concept of Residual Blocks. In this network, a technique called

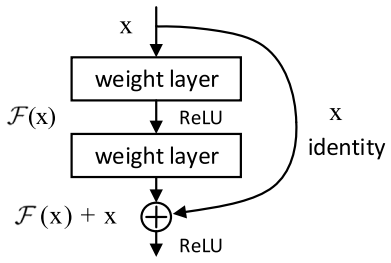


Fig. 1. Illustration of a single residual block with a shortcut connection used to maintain feature information from previous layers.

shortcut connection is developed. As shown in Fig. 1, the shortcut connection connects activations of a layer to further layers by skipping some layers in between. This forms a residual block. ResNets are made by stacking these residual blocks together. The idea behind this is to learn the residual mapping (the difference between the input and output of a layer) instead of the actual mapping, making it easier for the network to learn the identity function and thereby mitigating the vanishing gradient problem in deep networks. Therefore, instead of say $h(x)$, initial mapping, let the network fit:

$$\mathcal{F}(x) := h(x) - x, \tag{3.3}$$

The advantage of adding this type of shortcut connection is that if any layer hampers the performance of the architecture, it will be skipped by regularization. So, this results in training a very deep neural network without the problems caused by vanishing/exploding gradient. Therefore, the ResNet-V2 structure is employed as the network backbone. The approximated relationship is described as:

$$\mathbf{x}^{l+1} = \mathbf{x}^{l} + \mathcal{F}(\sigma(\mathbf{x}^{l}), \theta^{\mathbf{l}}), \tag{3.4}$$

where σ denotes the pre-activation, $\mathbf{x}^l$ and θ^l are the input feature and hyperparameters to the lth Residual Unit, respectively. In this way, the optimization is further eased and meantime, the pre-activation scheme can improve the regularization in order to reduce overfitting (He et al., 2016b).

An illustration of the ResNet-V2 block is illustrated in Fig. 2. The residual block has two convolutional layers with the same number of output channels. Each convolutional layer is preceded by a batch normalization layer and a ReLU activation function. Then, we skip these two convolution operations and add the input directly. This implies that the output of the two convolutional layers has to be of the same shape as the input, so that they can be added together. If we want to change the number of channels, we need to introduce an additional 1×1 convolutional layer to transform the input into the desired shape for the addition operation.

The typical single task learning process of a neural network is described as follows: suppose $\mathbf{x}$ represents the input domain (i.e., permeability fields) and $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2\}$ is the set of desired prediction domains (i.e., R_s and S_g maps). The aim is to learn the approximate relationship between input and prediction domains:

$$\mathcal{F}_{\mathbf{x}} = \left\{ \hat{f}_{\mathbf{xy}_1}, \hat{f}_{\mathbf{xy}_2} \right\}, \tag{3.5}$$

where, $\hat{f}_{\mathbf{xy}_1}$ and $\hat{f}_{\mathbf{xy}_2}$ outputs $\mathbf{y}_1$ and $\mathbf{y}_2$ given $\mathbf{x}$, respectively. The way of training $\hat{f}_{\mathbf{xy}_1}(x)$ is to find parameters that minimize the loss function, e.g., mean absolute error (MAE). As for $\hat{f}_{\mathbf{xy}_2}(x)$, the procedure is the same.

As shown in Fig. 3, ResNet is served as the network backbone and combines Encoder–Decoder architecture to form the single task learning framework in this paper, and the multi-task framework described later is also developed based on this framework. In particular, in order to capture temporal dynamics, we apply a three dimensional variation of Encoder–Decoder architecture which means the convolutional kernels are 3-D that can extract information in both the temporal and spatial dimensions (Maturana and Scherer, 2015).

3.3. Multi-task Learning (MTL)

Currently, most methods for multiphase flow are focused on only one of these tasks (i.e., saturation or pressure maps), and they also achieve the state-of-the-art performance through the technique of deep learning. However, there may be instances when learning from many related tasks at the same time would lead to better modeling performance (Brüggemann et al., 2021). This is addressed in the domain of multi-task learning, a subfield of machine learning in which multiple objectives are trained within the same model simultaneously. Compared to the single-task methods where each individual task is solved separately by its own network, recently, several multi-task learning methods in computer vision have shown a promising direction to improve the predictions by jointly tackling multiple tasks to boost for each other (Misra et al., 2016).

Compared to single-task learning, the loss function of typical multi-task learning is described by:

$$\mathcal{L} = \left| \hat{f}_{\mathbf{xy}_1}(x) - y_1 \right| + \left| \hat{f}_{\mathbf{xy}_2}(x) - y_2 \right|, \tag{3.6}$$

where $|\cdot|$ denote the MAE value which is also referred to as L_1 norm. Moreover, the first and the last terms are the standard losses for training $\hat{f}_{\mathbf{xy}_2}$ and $\hat{f}_{\mathbf{xy}_2}$, respectively.

MTL is used in many fields and joint learning is considered to give an improved representation than other STL since this method can capture the intrinsic association features between tasks (Vandenhende et al., 2021). In the context of deep learning, MTL is performed by learning shared representations from multiple supervisory tasks. Classical deep multi-task architectures were hard parameter sharing techniques which the parameter set is divided into shared and task-specific parameters, as shown in Fig. 4. MTL models typically consist of a shared encoder that branches out into task-specific heads. Multi-task learning takes care to fit all tasks, which is equivalent to regularization, thus avoiding overfitting a single task learning.

3.4. Multi-task Learning with Consistency (MTLC)

The predictions of R_s and S_g maps are important and challenging for CO_2 trapping in deep saline aquifers. The general objective of multi-task learning is to improve generalization by leveraging the domain-specific information contained in the training signals of related tasks. Common multi-task learning approaches are a shared feature extractor component (encoder) with multiple "heads" (decoder) that perform separate tasks. The fusion and sharing ways may utilize the correlative information between tasks, but there exist some drawbacks. For example, the integration of different features might result into ambiguity of information; the fusion does not explicitly model the task-level interaction where we do not know what information is transmitted (Zhang et al., 2019).

As we know, the learning tasks of R_s and S_g are predictions of different aspects of one underlying permeability field. Hence inconsistency among predictions implies contradiction and is inherently undesirable. Furthermore, these two parameters has the similarity pattern due to the same CO_2 trajectory. This similarity pattern between the related tasks are informative and can be used to better fit the data. The dynamic processes of R_s and S_g are connected by the underlying physics and consequently enforce some constrains on each other, referred to as consistency constraints (Zamir et al., 2020). In this work, the consistency constraints are learned form the data rather than an a prior given relationship, which makes the method applicable to any pairs of tasks that are not independent, particularly when their analytical relationship is unknown or difficult to formulate.

The independent single-task learning satisfies various ideal properties, if given infinite amount of labeled data. However, there is only limited amount of expensive labeled data in practice. Thus, a label efficient multi-task learning method is presented which introduces the concept of cross-task consistency. The difference between single task

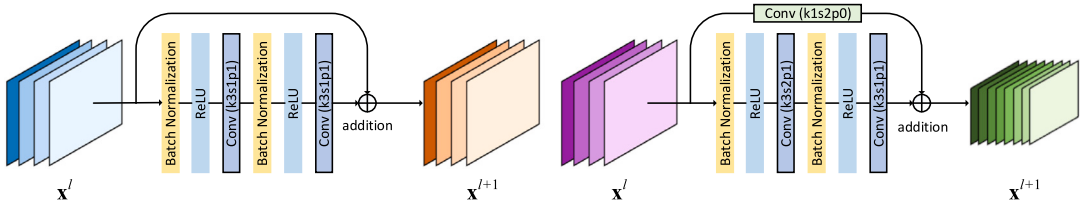


Fig. 2. Illustration of the basic residual blocks of ResNet-V2 without (left) and with (right) 1 × 1 convolution. The 1 × 1 convolution operation is used to transform the input into the desired size for the addition operation.

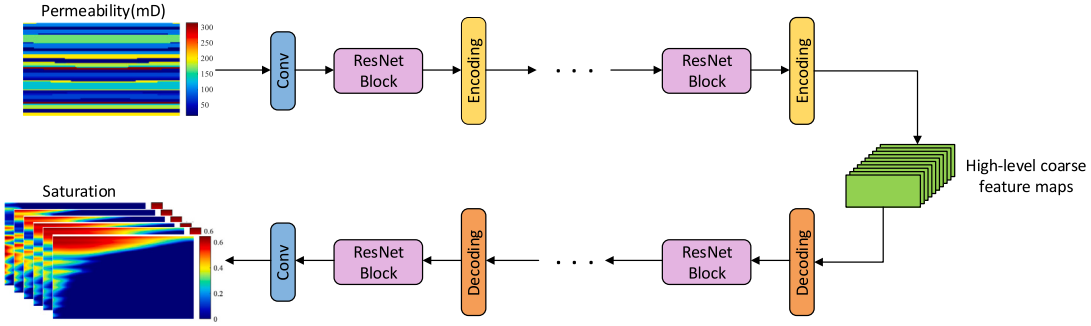


Fig. 3. The architecture for Single Task Learning (STL) network designed to predict gas saturation.

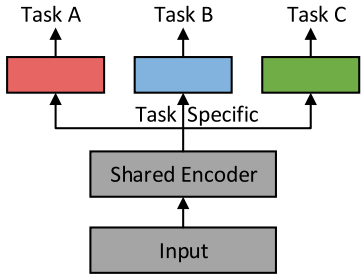


Fig. 4. Schematic of the general Multi-Task Learning (MTL) framework using deep neural networks for simultaneously learning multiple tasks. The shared encoder branches out to distinct decoders for each task, leveraging shared knowledge while allowing task-specific learning.

network, common MTL network and MTL with consistency (MTLC) is shown in Fig. 5. The MTLC consists of one shared encoder, two task-specific decoders, and an additional encoder–decoder pair for the transformation between R_s and S_g. The shared encoder is responsible for processing the permeability fields and generating a shared representation for both tasks. Then, the encoder branches out two decoders, one for each task. These decoders receive the shared representation and transform it into a task-specific output. Each decoder is specialized in generating output for its respective task. Finally, an additional encoder–decoder pair is designed to enforce consistency between tasks. The encoder in this pair takes the output S_g and generates a representation, which is then transformed into R_s by the corresponding decoder. This additional encoder–decoder pair helps in ensuring that the learned representations for S_g and R_s are consistent with each other. The detailed architecture of the encoder and decoder architecture is shown in Table 1. It is worth noting that the number of encoders and decoders varies between the single-task learning (STL), multi-task learning (MTL), and multi-task learning with consistency (MTLC) models. However, the layers within the encoder and decoder remain the same across all three models. This ensures that the architectural differences between the models are primarily related to the handling of tasks, while maintaining consistency in the internal structure of the encoders and decoders.

Therefore, an additional constrains to guide the training toward cross-task consistency is introduced. In addition to $\hat{f}_{\mathbf{xy}_1}$ and $\hat{f}_{\mathbf{xy}_2}$, the function $\mathcal{F}_{\mathbf{y}} = \left\{\hat{f}_{\mathbf{y}_1\mathbf{y}_2}\right\}$ is also defined which is the set of cross-task functions that map the prediction domains onto each other. The loss function for training the MTLC is defined as:

$$\mathcal{L}_{\mathbf{xy}_1\mathbf{y}_2} = \left|\hat{f}_{\mathbf{xy}_1}(x) - y_1\right| + \left|\hat{f}_{\mathbf{y}_1\mathbf{y}_2} \circ \hat{f}_{\mathbf{xy}_1}(x) - \hat{f}_{\mathbf{xy}_2}(x)\right| + \left|\hat{f}_{\mathbf{xy}_2}(x) - y_2\right|, \quad (3.7)$$

where, the middle term is the consistency term which enforces that predicting $\mathbf{y}_2$ out of the predicted $\mathbf{y}_1$ yields the same result as directly predicting $\mathbf{y}_2$ out of $\mathbf{x}$. This part of the network is aim to discover the similar patterns from the data. This consistency constraint helps the model generalize better by focusing on the common aspects of the tasks rather than learning task-specific features. It also helps regularize the model, preventing overfitting, and allowing it to exploit complementary information provided by both tasks. Therefore, the learning of predicting $\mathbf{y}_1$ and $\mathbf{y}_2$ are not independent anymore and a single deep learning framework involves simultaneous training of two tasks. In this work, the S_g and R_s maps are considered as $\mathbf{y}_1$ and $\mathbf{y}_2$, respectively.

4. Numerical experiments and results

4.1. Experiment setting

4.1.1. Datasets generation

In order to illustrate the performance of our method for predicting the CO_2 migration in deep saline aquifers, a 2-D CCS simulation system is considered. From a geological perspective, saline aquifers often have a dominant length in the longitudinal direction compared to their cross-sectional width. Since gravity plays an essential role in CO_2 trapping process, the 2D xz representation is preferred to xy representation in order to retain important vertical heterogeneity and fluid migration behavior. The model simulates the migration of CO_2 within a 2-D vertical cross-section aquifer domain. The dimensions are 200 m × 96 m, with a grid block size of 1 m × 1 m. The initial pressure is set to be 2.5×10^7 Pa with a constant temperature of 300 K. No flow boundary conditions are imposed on all sides of the aquifer. CO_2 is injected through the bottom 48 m of the domain on the left side with a constant rate (2×10^{-4} pore volumes per day). A production well is placed on the entire right side. The total simulation time and injection time is 600 days. The nonlinear relative permeability curves and capillary pressure

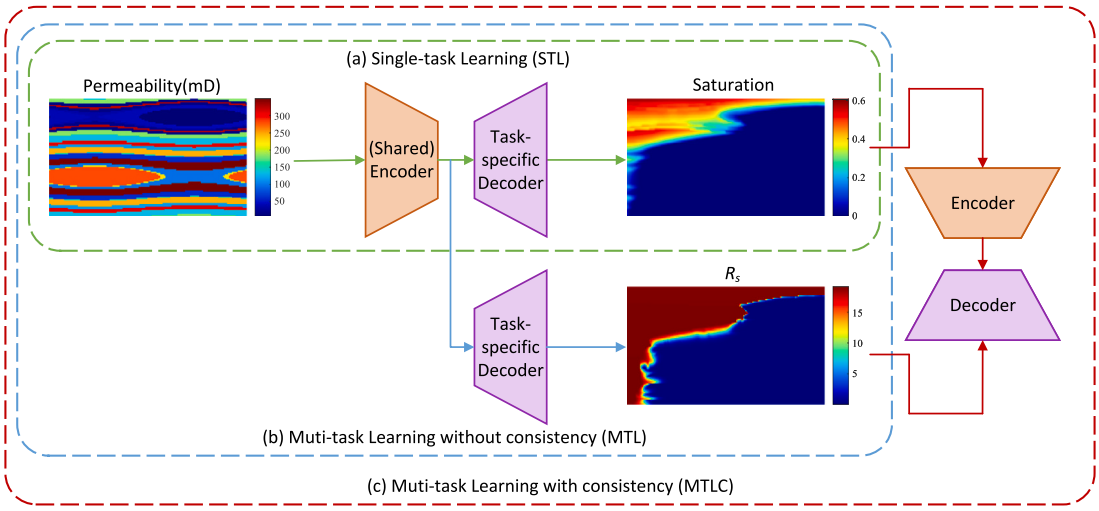


Fig. 5. Illustration of the differences between STL, MTL and MTLC. MTLC incorporates an additional encoder–decoder architecture to enforce cross-task consistency, enabling the learning process to better capture and utilize the shared underlying structures between tasks.

Table 1
Encoder–Decoder architecture details. k denotes the kernel size; s denotes the stride, c denotes the number of kernels; Conv denotes convolutional layer; Residual block is described in Fig. 2.

Net	Layer	Output size
Encoder	Input	(96,200,6,1)
	Conv, k = (3 × 3 × 3), s = (2 × 2 × 1), c = 16	(48,100,6,16)
	Residual block, k = (3 × 3 × 3), s = (1 × 1 × 2), c = 32	(48,100,3,32)
	Residual block, k = (3 × 3 × 3), s = (2 × 2 × 1), c = 64	(24,50,3,64)
	Residual block, k = (3 × 3 × 3), s = (1 × 1 × 2), c = 128	(24,50,2,128)
	Residual block, k = (3 × 3 × 3), s = (2 × 2 × 1), c = 256	(12,25,2,256)
	Residual block, k = (3 × 3 × 3), s = (1 × 1 × 1), c = 256	(12,25,2,256)
Decoder	Upsampling/Residual block, k = (3 × 3 × 3), s = (2 × 2 × 1), c = 128	(24,50,2,128)
	Upsampling/Residual block, k = (3 × 3 × 3), s = (1 × 1 × 2), c = 64	(24,50,3,64)
	Upsampling/Residual block, k = (3 × 3 × 3), s = (2 × 2 × 1), c = 32	(48,100,3,32)
	Upsampling/Residual block, k = (3 × 3 × 3), s = (1 × 1 × 2), c = 16	(48,100,6,16)
	Upsampling/Residual block, k = (3 × 3 × 3), s = (2 × 2 × 1), c = 8	(96,200,6,8)
	Conv, k = (3 × 3 × 3), s = (1 × 1 × 1), c = 1	(96,200,6,1)

Table 2
Physical parameters and simulation setup utilized in the DARSim numerical simulation.

Parameter	Value	Unit
Aquifer length	200	m
Aquifer height	96	m
Porosity	0.2	–
CO_2 injection rate	2×10^{-4}	Pore volumes per day
Initial pressure	2.5×10^7	Pa
Bottom hole pressure	2.5×10^7	Pa
Temperature (isothermal)	300	K
CO_2 density at STC	1.98	kg/m^3
Brine density at STC	1060	kg/m^3
Brine salinity	1×10^5	Parts per million

curves are shown in Fig. 6. Physical parameters and simulation settings are presented in Table 2.

To mimic the geological formations used for CO_2 sequestration, a total of 1400 permeability fields are generated using the open-source package Stanford Geostatistical Modeling Software (SGeMS) (Remy et al., 2009) with laterally correlated heterogeneity. The distribution of log-transformed permeability and one random realization of the permeability field are shown in Fig. 7. The forward simulation is performed using DARSim and we collect the output state maps (R_s and S_g maps) at prescribed time steps respectively. Each simulation takes 4800 s on an Intel Core i7-12700K.

We are interested in the spatial–temporal evolution of the R_s and S_g maps during CO_2 injection period. Thus, we collect the output state maps at six uniform time instances to train the networks. The goal is to train networks with the simulation data that yield reliable predictions of R_s and S_g on the hitherto unseen permeability fields in the test set.

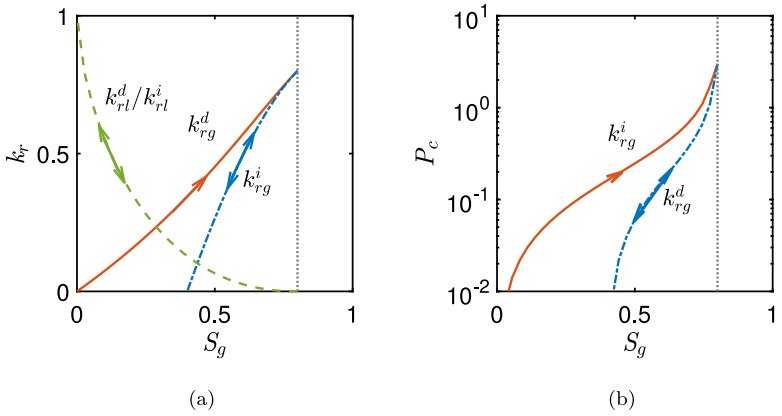
4.1.2. Training procedures

The loss function used for the training models is the Mean Absolute Error (MAE), defined as:

$$\mathcal{L}_{\text{MAE}} = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \left\| \hat{\mathbf{y}}_i - \mathbf{y}_i \right\|_1 , \tag{4.1}$$

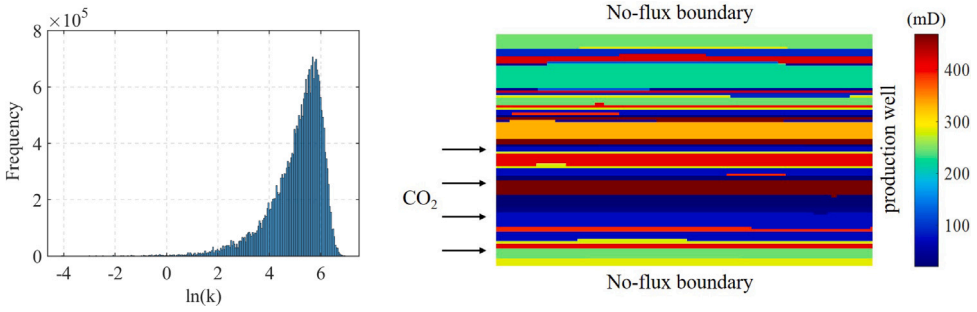
where N is the total number of geo-models in the training set. In the training process, the loss function is minimized by tuning the network parameters θ. The gradient of the loss function with respect to θ is automatically computed by back-propagation (Hecht-Nielsen, 1992). In this work, the batch size is set to 16, and the adaptive moment estimation (Adam) optimization algorithm (Kingma and Ba, 2014) is used with the initial learning rate and learning rate decay set to 0.001 and 0.1, respectively. This has been found to be an effective procedure for the training of many deep neural network architectures. Moreover, we utilize varying sizes of training datasets with 300, 600, 900, and 1200 permeability realizations to train the models. A separate, fixed set of 200 permeability realizations, not included in the training data, is used for testing the performance of all trained models. Due to the limited size of the dataset, we have not employed a separate validation dataset.

During the training process, the MAE metric is used to monitor the convergence of both the training and test errors. Additionally,



(a) (b)

Fig. 6. (a) Primary drainage and imbibition curves for liquid and gas phases, where the superscripts d and i represent drainage and imbibition, respectively. A single-headed arrow indicates that the process along a given curve is irreversible, while a double-headed arrow signifies that the process is reversible. (b) Illustration of drainage/imbibition capillary pressure curves.



(a) (b)

Fig. 7. (a) The histogram of the 1400 log-permeability fields used in both training and test datasets. (b) Schematic of well configurations and boundary conditions for a layered heterogeneous system.

to evaluate the quality of the trained models, we also consider two commonly used metrics, the root mean square error (RMSE) metric and the coefficient of determination (R^2) evaluated on 200 prefixed test samples. The RMSE is calculated by:

$$\text{RMSE} = \sqrt{\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|_2^2}, \tag{4.2}$$

The R^2 is defined as:

$$R^2 = 1 - \frac{\sum_{i=1}^{N_{\text{test}}} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|_2^2}{\sum_{i=1}^{N_{\text{test}}} \|\mathbf{y}_i - \bar{\mathbf{y}}_i\|_2^2}, \tag{4.3}$$

In particular, smaller MAE and RMSE values represent better performance, and an R^2 value closer to 1 indicates better prediction quality.

4.2. Performance evaluation

4.2.1. Training and testing results

Fig. 8 shows the MAE value of MTLC training and testing loss with the number of epochs and for training ensemble sizes of 300, 600, 900, and 1200 realizations. The models are trained on a NVIDIA GeForce GTX 3080 GPU which requires around 2000–7000 s for training 300 epochs when the training sample size varies from 300 to 1200. The training data generation and training times for the deep learning models are part of the initial setup process and need only be considered once. After training, given a new realization, the MTLC can provide predictions for the state maps, at 6 time steps, in an elapsed time of about 19 ms. This speed-up is 5 orders of magnitude faster than the numerical simulator, which is attributed to the benefits of using a GPU for the deep learning model's execution. The GPU's highly parallelizable nature and the extensive support provided by modern deep learning frameworks significantly contribute to the efficiency advantage of the MTLC model over the traditional numerical simulator. Therefore, MTLC model becomes more beneficial when a large number of realizations for uncertainty quantification or optimization. It is observed that the MAE starts to stabilize after 200 epochs for both R_s and S_g. For the testing dataset, the MAE for R_s is approximately 0.02 and for S_g is 0.005. The error for R_s is relatively large due to the presence of response discontinuity, which is a well-known challenge for other surrogate models, and this error can be considered acceptable in the context of CCS.

The performance of our MTLC model in approximating the time-dependent multi-output is further demonstrated in Figs. 9 and 10, which depict a comparison of the R_s and S_g fields at various time instances (100, 200, 300, 400, 500 and 600 days) predicted by DARSim and our MTLC model using 1200 training samples. As expected, the model achieves high approximation accuracy for both R_s and S_g fields over time.

4.2.2. Performance comparison

To demonstrate the efficiency and effectiveness of the MTLC framework, we compare the performance of MTLC, MTL and STL. Each model is trained using different numbers of training samples, 300, 600, 900, 1200 realizations, respectively. After training, the prediction times for STL, MTL, MTLC are 14 ms, 17 ms and 19 ms, respectively, with negligible differences. It is worth noting that for STL, the model evaluation needs to be performed separately for R_s and S_g, while MTL and MTLC only need to be done once. The R^2 scores for the test dataset

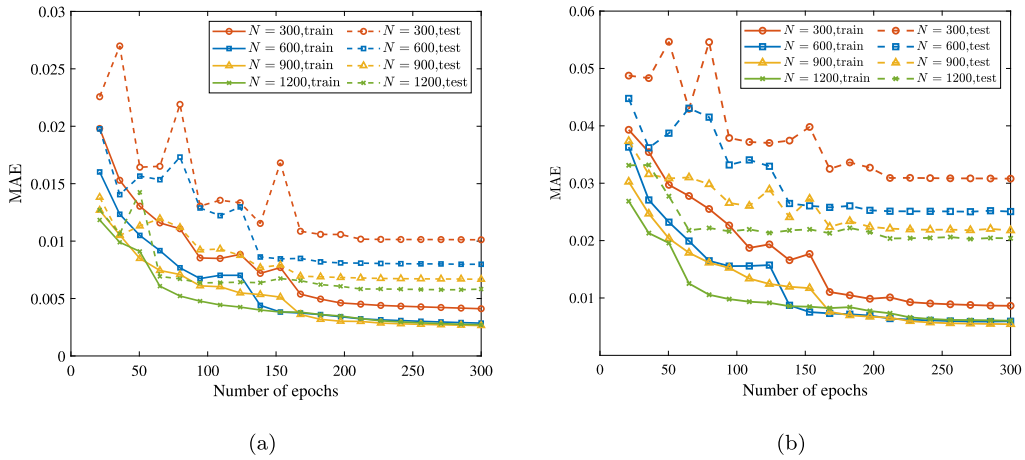


Fig. 8. (a) Evolution of Mean Absolute Error (MAE) with the number of epochs in both the training and test sets for the task S_g, using different training sample sizes (N = 300, 600, 900 and 1200). (b) Evolution of MAE with the number of epochs in both the training and test sets for the task R_s, using different training sample sizes.

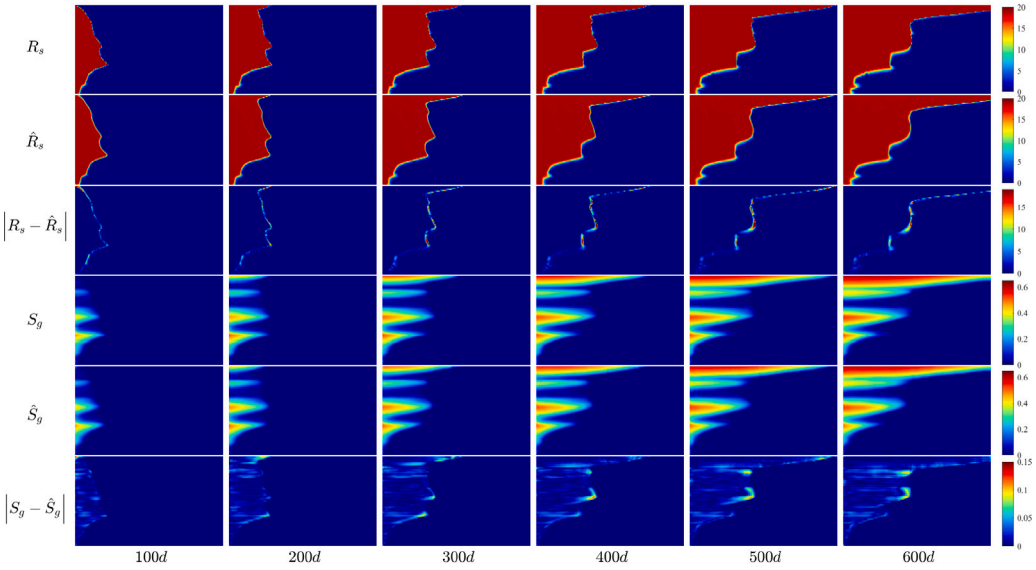


Fig. 9. Example 1: Snapshots of the R_s and S_g fields at several time instances obtained from DARSim and the corresponding $\hat{R}_s$ and $\hat{S}_g$ predicted by the MTLC model trained on 1200 samples, demonstrating the high accuracy of the MTLC model in approximating the dynamic states of these fields.

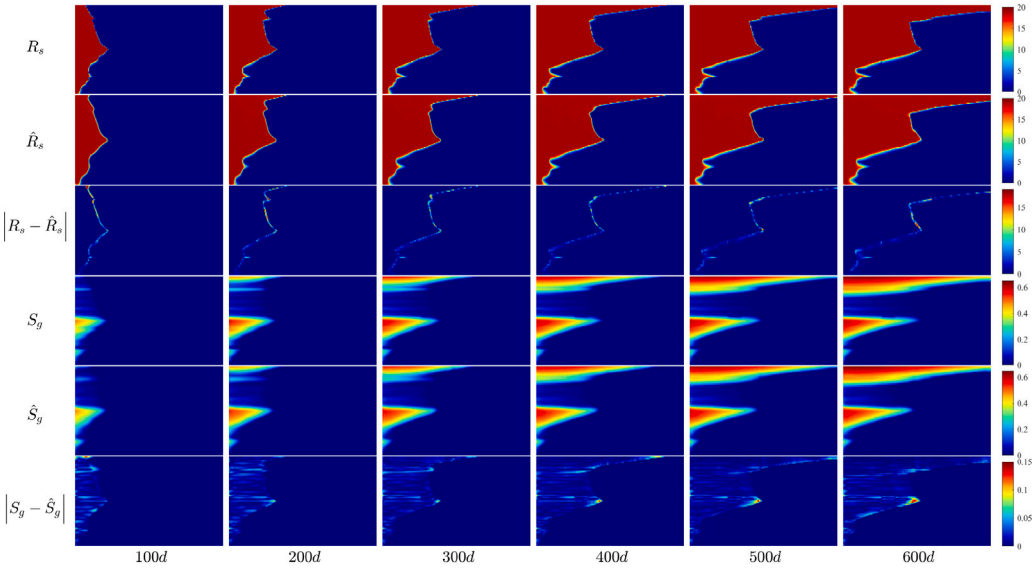


Fig. 10. Example 2: Another set of snapshots of the R_s and S_g fields at several time instances obtained from DARSim and the corresponding $\hat{R}_s$ and $\hat{S}_g$ predicted by the MTLC model using 1200 training samples, further validating the effectiveness of the MTLC model.

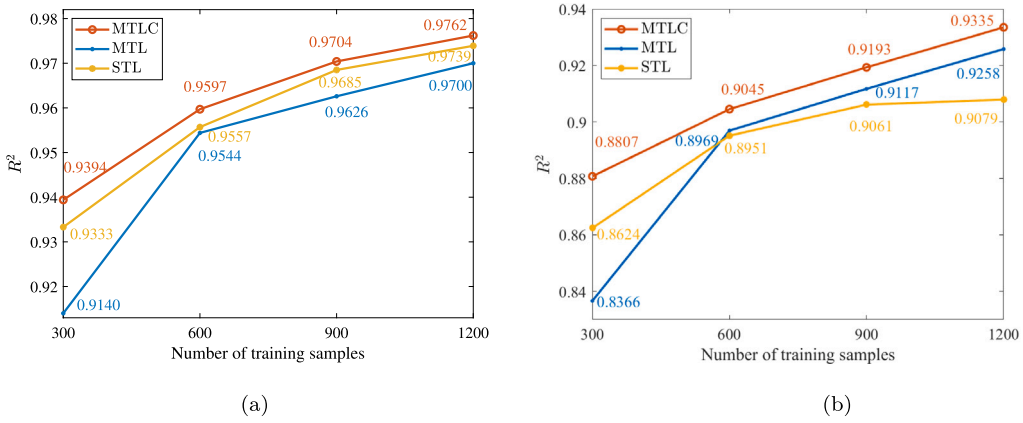


Fig. 11. Comparison of R^2 scores for S_g from networks trained using different methods (STL, MTL, MTLC), evaluated on 200 test samples with varying numbers of training samples. (b) Comparison of R^2 scores for R_s from networks trained using different methods (STL, MTL, MTLC), evaluated on 200 test samples with varying numbers of training samples.

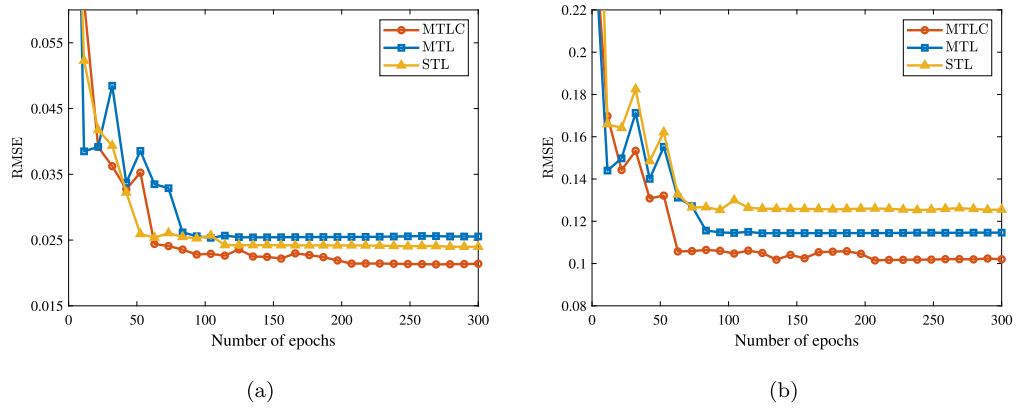


Fig. 12. (a) Illustration of RMSE decay with the number of epochs for STL, MTL, and MTLC on the task of predicting S_g using 1200 training samples. (b) Illustration of RMSE decay with the number of epochs for STL, MTL, and MTLC on the task of predicting R_s using 1200 training samples.

for each model with different training data set sizes are shown in Fig. 11. The figure shows that the model achieves a relatively high R^2 value of 0.9394 for the S_g problem and 0.8807 for R_s with only 300 training samples, even with high input dimensions and the presence of response discontinuity. These values are much higher than those of the STL (0.9140 and 0.8366). When increasing the samples size to 1200, the model achieves R^2 values of 0.9762 and 0.9335 for S_g and R_s, respectively. We also performed the R^2 score test only for the plume grid cells which have non-zero values in the reference maps from the numerical simulator, to demonstrate the significance of the difference. The R^2 scores for the plume area corresponding to R_s are 0.8616, 0.8226, and 0.7876 for MTLC, MTL, and STL, respectively. Meanwhile, the R^2 scores for S_g are 0.9574, 0.9401, and 0.9488, respectively. It is also observed that the performance of MTL for R_s is better than STL, while that for S_g is worse than STL. This phenomenon proves that common hard parameters sharing scheme could compromise the performance of any task. In contrast, the MTLC framework improves the performance of both tasks due to the consistency constraints.

The evolutions of testing RMSE for three networks trained with 1200 training samples are shown in Fig. 12. It is evident that the MTLC has a higher convergence speed for R_s. We also observe that the results of MTLC outperform those of the other models for both tasks.

In Fig. 13, we present an example of the predictions in the test dataset at 600 days. The MTLC framework not only achieves the best performance compared to STL and MTL, but it also provides the more accurate positions of the R_s and S_g fronts, making it a more reliable model for predicting CO_2 saturation and CO_2-brine ratio in the context of CCS projects. The accuracy of the MTLC framework is further quantified by calculating the quantities of injected CO_2 trapped by dissolution trapping and residual trapping for the test case shown in Fig. 13. As depicted in Fig. 14, it is evident that the MTLC framework provides the most consistent predictions for both dissolution and residual trapped CO_2 quantities when compared to the results from the numerical simulator. This highlights the effectiveness of the MTLC framework for predicting CO_2 trapping in CCS projects.

4.2.3. Uncertainty modeling

In this study, we compared the performance of three different network structures – STL, MTL, and MTLC – in an uncertainty quantification task focused on estimating R_s and S_g. Fig. 15 displays the estimated distributions of 200 test permeability realizations for each method at location (48 m, 100 m) after 600 days. The probability density functions (PDFs) of S_g and R_s obtained using the MTLC framework were nearly indistinguishable from those obtained using the numerical simulator. This result highlights the effectiveness of the MTLC model in handling uncertainty quantification tasks within the CCS domain, demonstrating its potential for practical application in similar contexts.

The results presented in this study demonstrate that the proposed MTLC method can provide accurate solutions for dynamic CCS simulation in heterogeneous saline aquifers. Therefore, it is able to be regarded as a data augmentation technique, being able to improve the accuracy of predictions with limited data.

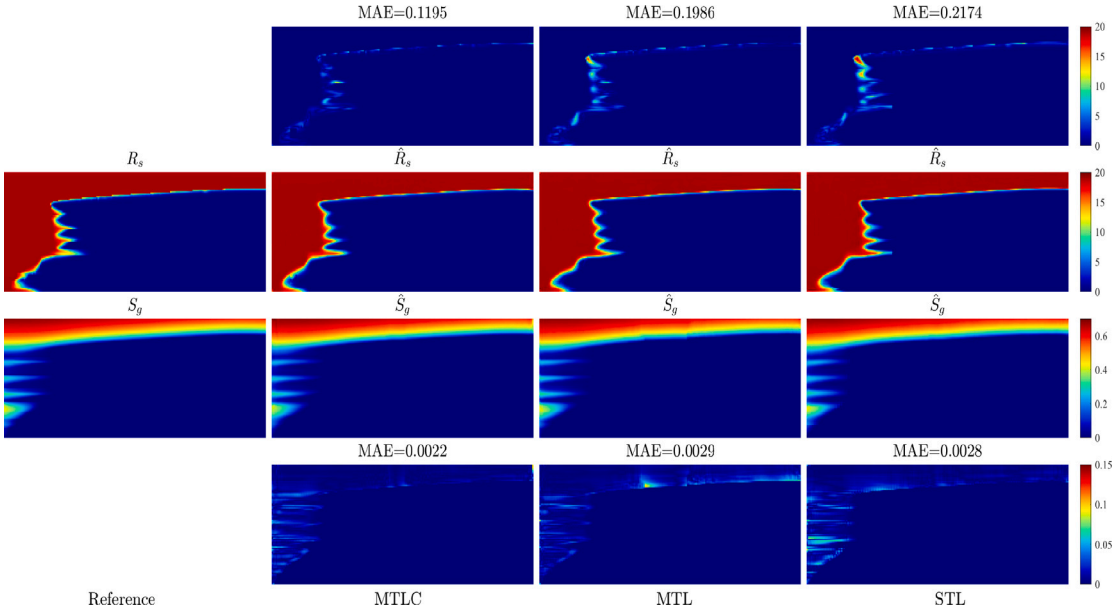


Fig. 13. Test results for the three models (STL, MTL, and MTLC) at 600 days. The top and bottom rows present the absolute error with respect to the reference, demonstrating the high accuracy of the MTLC model.

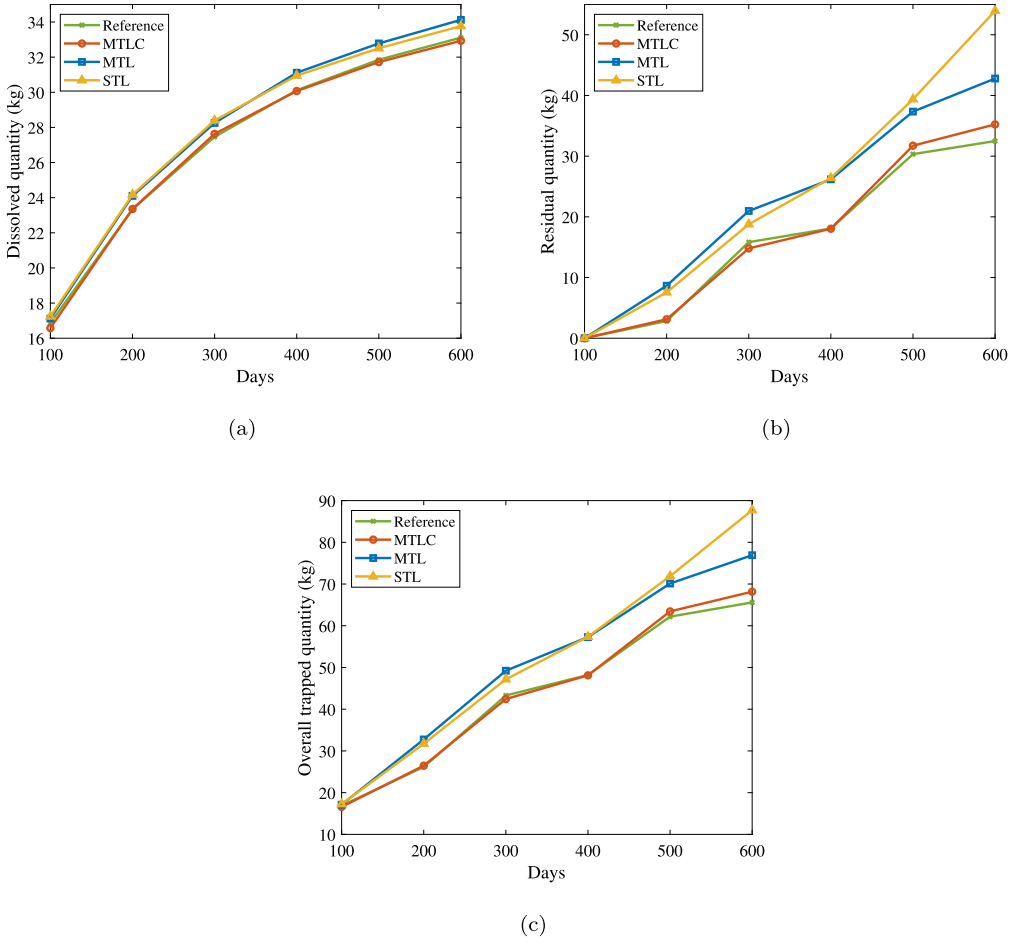


Fig. 14. Comparison of the predicted amount of trapping by the numerical simulator (reference – green lines) and three models – MTLC (red lines), MTL (blue lines), and STL (yellow lines). (a) Dissolution trapping. (b) Residual trapping. (c) Overall trapping.

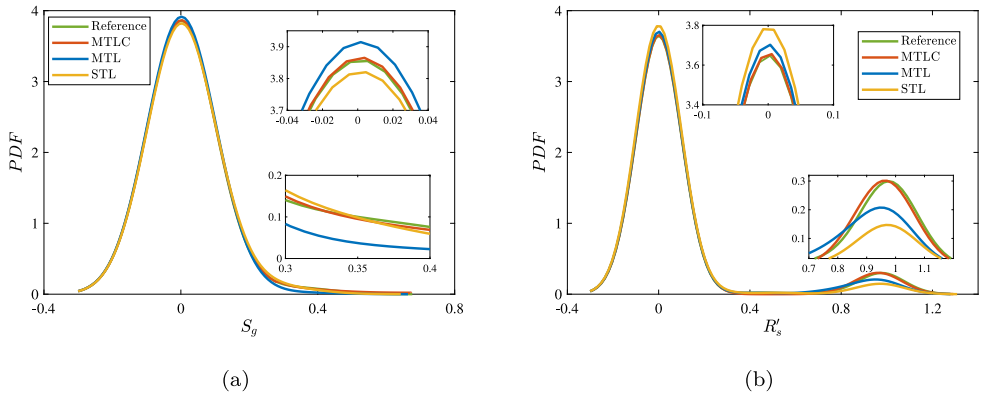


Fig. 15. (a) Probability density functions (PDF) of S_g at the specific location (48 m, 100 m) at 600 days, estimated by numerical simulator and the three model variations (STL, MTL, MTLC). (b) PDFs of R_s at the same location and time point.

5. Discussions and conclusions

In this study, we presented a novel multi-task learning with consistency (MTLC) framework for augmenting learning the CO_2 gas saturation S_g and CO_2-brine ratio R_s during the storage dynamic process in saline aquifers. Our approach leverages the encoder–decoder architectures to achieve cross-task consistency, which enforces constraints on each task and facilitates the simultaneous prediction of these two related tasks. Therefore, it can serve as a field-scale applicable alternative to conventional simulators for CO_2 storage management.

We compared our MTLC framework to traditional deep learning methods in dynamic CCS simulations in deep saline aquifers with heterogeneous formations, demonstrating its effectiveness. Our evaluations revealed that incorporating cross-task consistency led to improved data fitting, more accurate predictions and better generalization in the models.

However, CO_2 plume migration in deep saline aquifers is a complex multicomponent multiphase problem controlled by the interplay of various mechanisms. Despite the excellent results, there are limitations that need to addressed in future work: (1) Our work simplifies certain aspects for numerical simulations in order to reduce complexity, such as neglecting molecular diffusion–dispersion. While these choices reduce complexity and computational time, accounting for more complex physics allows for a more comprehensive representation of the reservoir. (2) The deep learning model is trained based on physical trapping mechanisms, which do not capture the full complexity of thermo-hydro-mechanical–chemical processes involved in geological CO_2 sequestration. This limitation should be considered when applying the model to real-world scenarios. (3) Future work could explore the extension to incorporate full 3D models to enhance the applicability in various scenarios potentially. (4) It is also critical to note that as the model's complexity increases, the size of the training dataset needed for accurate predictions could also increase. As a result, future work could explore ways to optimize training dataset size in the context of more complex real-world applications.

CRediT authorship contribution statement

Mengjie Zhao: Conceptualization, Methodology, Writing – original draft. **Yuhang Wang:** Conceptualization, Data curation, Writing – review & editing. **Marc Gerritsma:** Supervision, Writing – review & editing. **Hadi Hajibeygi:** Supervision, Funding acquisition, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

MZ acknowledges the China Scholarship Council for supporting her PhD work at TU Delft. YW was supported by the "CUG Scholar" Scientific Research Funds at China University of Geosciences (Wuhan) (Project No. 2022157). HH was partly funded by the Dutch National Science Foundation (NWO) under the Talent Scheme Vidi, project ADMIRE.

References

Albawi, S., Mohammed, T.A., Al-Zawi, S., 2017. Understanding of a convolutional neural network. In: 2017 International Conference on Engineering and Technology (ICET). IEEE, pp. 1–6.

Babaei, M., Pan, I., 2016. Performance comparison of several response surface surrogate models and ensemble methods for water injection optimization under uncertainty. Comput. Geosci. 91, 19–32.

Badrinarayanan, V., Kendall, A., Cipolla, R., 2017. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 39 (12), 2481–2495.

Brüggemann, D., Kanakis, M., Obukhov, A., Georgoulis, S., Van Gool, L., 2021. Exploring relational context for multi-task dense prediction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 15869–15878.

Celia, M., Bachu, S., Nordbotten, J., Bandilla, K., 2015. Status of CO_2 storage in deep saline aquifers with emphasis on modeling approaches and practical simulations. Water Resour. Res. 51 (9), 6846–6892.

Chen, G., Zhang, K., Xue, X., Zhang, L., Yao, J., Sun, H., Fan, L., Yang, Y., 2020a. Surrogate-assisted evolutionary algorithm with dimensionality reduction method for water flooding production optimization. J. Pet. Sci. Eng. 185, 106633.

Chen, G., Zhang, K., Zhang, L., Xue, X., Ji, D., Yao, C., et al., 2020b. Global and local surrogate-model-assisted differential evolution for waterflooding production optimization. SPE J. 25 (01), 105–118.

Christelis, V., Regis, R.G., Mantoglou, A., 2018. Surrogate-based pumping optimization of coastal aquifers under limited computational budgets. J. Hydroinform. 20 (1), 164–176.

Chu, A.K., Benson, S.M., Wen, G., 2023. Deep-learning-based flow prediction for CO_2 storage in shale–sandstone formations. Energies 16 (1), 246.

Cusini, M., van Kruijsdijk, C., Hajibeygi, H., 2016. Algebraic dynamic multilevel (ADM) method for fully implicit simulations of multiphase flow in porous media. J. Comput. Phys. 314, 60–79.

Elenius, M., Voskov, D., Tchelepi, H., 2015. Interactions between gravity currents and convective dissolution. Adv. Water Resour. 83, 77–88.

Gunter, W., Perkins, E., Hutcheon, I., 2000. Aquifer disposal of acid gases: modelling of water–rock reactions for trapping of acid wastes. Appl. Geochem. 15 (8), 1085–1095.

Guo, Z., Reynolds, A.C., 2018. Robust life-cycle production optimization with a support-vector-regression proxy. Spe J. 23 (06), 2409–2427.

Hamdi, H., Couckuyt, I., Sousa, M.C., Dhaene, T., 2017. Gaussian processes for history-matching: application to an unconventional gas reservoir. Comput. Geosci. 21 (2), 267–287.

He, K., Zhang, X., Ren, S., Sun, J., 2016a. Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 770–778.

He, K., Zhang, X., Ren, S., Sun, J., 2016b. Identity mappings in deep residual networks. In: Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, the Netherlands, October 11–14, 2016, Proceedings, Part IV 14. Springer, pp. 630–645.

Hecht-Nielsen, R., 1992. Theory of the backpropagation neural network. In: Neural Networks for Perception. Elsevier, pp. 65–93.

Ide, S.T., Jessen, K., Orr Jr., F.M., 2007. Storage of CO_2 in saline aquifers: Effects of gravity, viscous, and capillary forces on amount and timing of trapping. Int. J. Greenh. Gas Control 1 (4), 481–491.

Kadeethum, T., O'Malley, D., Fuhg, J.N., Choi, Y., Lee, J., Viswanathan, H.S., Bouklas, N., 2021. A framework for data-driven solution and parameter estimation of pdes using conditional generative adversarial networks. Nat. Comput. Sci. 1 (12), 819–829.

Kamrava, S., Sahimi, M., Tahmasebi, P., 2021. Simulating fluid flow in complex porous materials by integrating the governing equations with deep-layered machines. Npj Comput. Mater. 7 (1), 127.

Kingma, D.P., Ba, J., 2014. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

Kopp, A., Binning, P.J., Johannsen, K., Helmig, R., Class, H., 2010. A contribution to risk analysis for leakage through abandoned wells in geological CO_2 storage. Adv. Water Resour. 33 (8), 867–879.

Krevor, S., de Coninck, H., Gasda, S., Ghaleigh, N.S., de Gooyert, V., Hajibeygi, H., Juanes, R., Neufeld, J., Roberts, J., Swennenhuis, F., 2023. Algebraic dynamic multilevel method for compositional flow in heterogeneous porous media. Nat. Rev. Earth Environ. 4, 102–118.

Liu, S., Johns, E., Davison, A.J., 2019. End-to-end multi-task learning with attention. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1871–1880.

Liu, D., Li, Y., Agarwal, R., 2020. Evaluation of CO_2 storage in a shale gas reservoir compared to a deep saline aquifer in the ordos basin of China. Energies 13 (13), 3397.

Maturana, D., Scherer, S., 2015. Voxnet: A 3D convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp. 922–928.

Misra, I., Shrivastava, A., Gupta, A., Hebert, M., 2016. Cross-stitch networks for multi-task learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3994–4003.

Mo, S., Zhu, Y., Zabaras, N., Shi, X., Wu, J., 2019. Deep convolutional encoder-decoder networks for uncertainty quantification of dynamic multiphase flow in heterogeneous media. Water Resour. Res. 55 (1), 703–728.

Redmon, J., Divvala, S., Girshick, R., Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 779–788.

Remy, N., Boucher, A., Wu, J., 2009. Applied Geostatistics with SGeMS: A User's Guide. Cambridge University Press.

Ren, S., He, K., Girshick, R., Sun, J., 2015. Faster R-CNN: Towards real-time object detection with region proposal networks. Adv. Neural Inf. Process. Syst. 28.

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18. Springer, pp. 234–241.

Rubin, E., De Coninck, H., 2005. IPCC special report on carbon dioxide capture and storage. Cambridge University Press TNO, UK, (2004): Cost Curves for CO2 Storage, Part 2005;2:14.

Selma, L., Seigo, O., Dohle, S., Siegrist, M., 2014. Public perception of carbon capture and storage (CCS): A review. Renew. Sustain. Energy Rev. 38, 848–863.

Simonyan, K., Zisserman, A., 2014. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

Tang, M., Ju, X., Durlofsky, L.J., 2022. Deep-learning-based coupled flow-geomechanics surrogate model for CO_2 sequestration. Int. J. Greenh. Gas Control 118, 103692.

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.

Vandenhende, S., Georgoulis, S., Van Gansbeke, W., Proesmans, M., Dai, D., Van Gool, L., 2021. Multi-task learning for dense prediction tasks: A survey. IEEE Trans. Pattern Anal. Mach. Intell. 44 (7), 3614–3633.

Wang, N., Chang, H., Zhang, D., 2021. Efficient uncertainty quantification and data assimilation via theory-guided convolutional neural network. SPE J. 26 (06), 4128–4156.

Wang, Y., HosseiniMehr, M., Marelis, A., Hajibeygi, H., 2023. A generic framework for multiscale simulation of high and low enthalpy fractured geothermal reservoirs under varying thermodynamic conditions. Energies 16 (2), 928.

Wang, Y., Vuik, C., Hajibeygi, H., 2022. Analysis of hydrodynamic trapping interactions during full-cycle injection and migration of CO_2 in deep saline aquifers. Adv. Water Resour. 159, 104073.

Wen, G., Hay, C., Benson, S.M., 2021a. CCSNet: a deep learning modeling suite for CO_2 storage. Adv. Water Resour. 155, 104009.

Wen, G., Tang, M., Benson, S.M., 2021b. Towards a predictor for CO_2 plume migration using deep neural networks. Int. J. Greenh. Gas Control 105, 103223.

Yan, B., Chen, B., Harp, D.R., Jia, W., Pawar, R.J., 2022. A robust deep learning workflow to predict multiphase flow behavior during geological CO_2 sequestration injection and post-injection periods. J. Hydrol. 607, 127542.

Zamir, A.R., Sax, A., Cheerla, N., Suri, R., Cao, Z., Malik, J., Guibas, L.J., 2020. Robust learning through cross-task consistency. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11197–11206.

Zhang, Z., Cui, Z., Xu, C., Yan, Y., Sebe, N., Yang, J., 2019. Pattern-affinitive propagation across depth, surface normal and semantic segmentation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4106–4115.

Zhao, M., Zhang, K., Chen, G., Zhao, X., Yao, C., Sun, H., et al., 2020a. A surrogate-assisted multi-objective evolutionary algorithm with dimension-reduction for production optimization. J. Pet. Sci. Eng. 192, 107192.

Zhao, M., Zhang, K., Chen, G., Zhao, X., Yao, J., Yao, C., et al., 2020b. A classification-based surrogate-assisted multiobjective evolutionary algorithm for production optimization under geological uncertainty. SPE J. 25 (05), 2450–2469.

Zhong, Z., Sun, A.Y., Jeong, H., 2019. Predicting CO_2 plume migration in heterogeneous formations using conditional deep convolutional generative adversarial network. Water Resour. Res. 55 (7), 5830–5851.

Zhu, Y., Zabaras, N., 2018. Bayesian deep convolutional encoder–decoder networks for surrogate modeling and uncertainty quantification. J. Comput. Phys. 366, 415–447.

Zhu, Y., Zabaras, N., Koutsourelakis, P.-S., Perdikaris, P., 2019. Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data. J. Comput. Phys. 394, 56–81.